Lecture Handout

Data Warehousing

Lecture No. 01

<u>Why Data Warehousing?</u>

- The world is changing (actually changed), either change or be left behind.
- Missing the opportunities or going in the wrong direction has prevented us from growing.
- What is the right direction?
- Harnessing the data, in a knowledge driven economy.

The world economy has moved form the industrial age into information driven knowledge economy. The information age is characterized by the computer technology, modern communication technology and Internet technology; all are popular in the world today. Governments around the globe have realized potential of information, as a “multi-factor” in the development of their economy, which not only creates wealth for the society, but also affects the future of the country. Thus, many countries in the world have placed the modern information technology into their strategic plans. They regard it as the most important strategic resource for the development their society, and are trying their best to reach and occupy the peak of the modern information driven knowledge economy.

<u>What is the right direction?</u>
Ever since the IT revolution that happened more than a decade ago every government has been trying and tried to increase our software exports. But have persistently failed to get the desired results. I happened to meet a gentleman who got venture capital of several million US dollars and I asked him why our software export has not gone up? His answer was simple, “we have been investing in outgoing or outdated tools and technologies”. We have also been just following India, without thinking for a moment, what India is today, started maybe a decade ago. So my next question was “what should we be doing today?” His answer was “we have captured and stored data for a long time, now it is time to explore and make use of that data”. There is a saying that “a fool and his money are soon parted”, since that gentleman was rich and is still rich, hence he does qualify to be a wise man, and his words of wisdom to be paid attention to.

<u>The Need for a Data Warehouse</u>

“Drowning in data and starving for information”

“Knowledge is power, Intelligence is absolute power!”

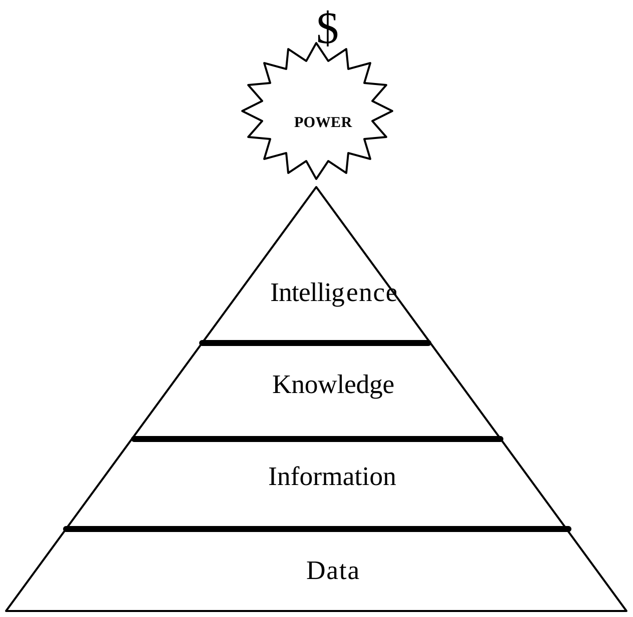


Figure-1.1: Relationship between Data, Information, Knowledge & Intelligence

Data is defined as numerical or other facts represented or recorded in a form suitable for processing by computers. Data is often the record or result of a transaction or an operation that involves modification of the contents of a database or insertion of rows in tables. Information in its simplest form is processed data that is meaningful. By processing, summarizing or analyzing data, organizations create information. For example the current balance, items sold, money made etc. This information should be designed to increase the knowledge of the individual, therefore, ultimately being tailored to the needs of the recipient. Information is processed data so that it becomes useful and provides answers to questions such as "who", "what", "where", and "when". Knowledge, on the other hand is an application of information and data, and gives an insight by answering the “how” questions. Knowledge is also the understanding gained through experience or study. Intelligence is appreciation of "why", and finally wisdom (not shown in the figure-1.1) is the application of intelligence and experience toward the attainment of common goals, and wise people are powerful. Remember knowledge is power.

Historical Overview

It is interesting to note that DSS (Decision Support System) processing as we know it today has reached this point after a long and complex evolution, and yet it continues to evolve. The origin of DSS goes back to the very early days of computers.

Figure-1.2 shows the historical overview or the evolution of data processing from the early 1960s upto 1980s. In the early 1960s, the world of computation consisted of exclusive applications that were executed on master files. The applications featured reports and programs, using languages like COBOL and punched cards i.e. the COBOLian era. The master files were stored on magnetic tapes, which were good for storing a large volume of data cheaply, but had the drawback of needing to be accessed sequentially, and being very unreliable (ask your system administrator even today about tape backup reliability). Experience showed that for a single pass of a magnetic tape that scanned 100% of the records, only 5% of the records, sometimes even less were actually required. In addition, reading an entire tape could take anywhere from 20-30 minutes, depending on the data and the processing required.

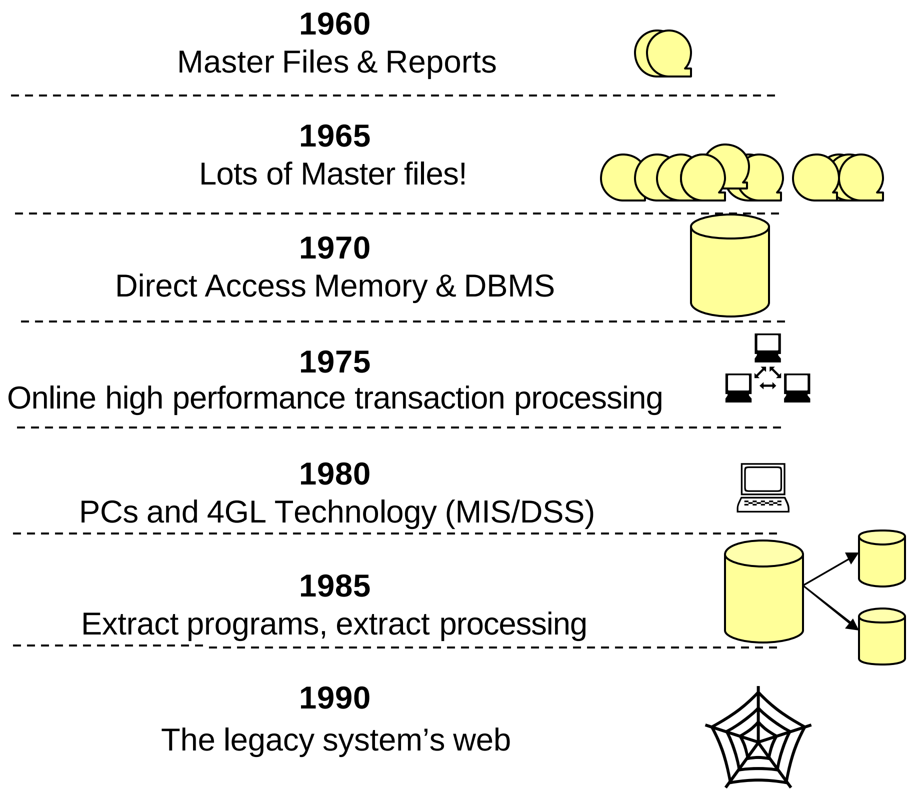

1960
Master Files & Reports
1965
Lots of Master files!
1970
Direct Access Memory & DBMS
1975
Online high performance transaction processing
1980
PCs and 4GL Technology (MIS/DSS)
1985
Extract programs, extract processing
1990
The legacy system's web

Figure-1.2: Historical Overview of use of Computers for Data Processing

Around the mid-1960s, the growth of master files and magnetic tapes exploded. Soon master files were used at every computer installation. This growth in usage of master files, resulted in huge amounts of redundant data. The spreading of master files and massive redundancy of data presented some very serious problems, such as:

- Data coherency i.e. the need to synchronize data upon update.
- Program maintenance complexity.
- Program development complexity.
- Requirement of additional hardware to support many tapes.

In a nut-shell, the inherent problems of master files because of the limitations of the medium used started to become a bottleneck. If we had continued to use only the magnetic tapes, we may not have had an Information revolution! Consequently, there would have never been large, fast MIS (Management Information Systems) systems, ATM systems, Airline Flight reservation systems, maybe not even Internet as we know it. As one of my teachers very rightly said, "every problem is an opportunity" therefore, the ability to store and manage data on diverse media (other than magnetic tapes) opened up the way for a very different and more powerful type of processing i.e. bringing the IT and the business user together as never before.

The advent of DASD

By 1970s, a new technology for the storage and access of data had had been introduced. The 1970s saw the advent of disk storage, or DASD (Direct Access Storage Device). Disk storage was fundamentally different from magnetic tape storage in the sense that data could be accessed directly on DASD i.e. non-sequentially. There was no need to go all the way through records 1, 2, 3, . . . k so as to reach the record k + 1. Once the address of record k + 1 was known, it was a simple matter to go to record k + 1 directly. Furthermore, the time required to go to record k + 1 was significantly less than the time required to scan a magnetic tape. Actually it took milliseconds to locate a record on a DASD i.e. orders of magnitude better performance than the magnetic tape.

With DASD came a new type of system software known as a DBMS (Data Base Management System). The purpose of the DBMS was to facilitate the programmer to store and access data on DASD. In addition, the DBMS took care of such tasks as storing data on DASD, indexing data, accessing it etc. With the winning combination of DASD and DBMS came a technological solution to the problems of magnetic tape based master files. When we look back at the mess that was created by master files and the mountains of redundant data aggregated on them, it is no wonder that database is defined as a single source of data for all processing and a prelude to a data warehouse i.e. "a single source of truth".

PC & 4GL

By the 1980s, more and new hardware/software, such as PCs and 4GLs (4th Generation Languages) began to come out. The end user began to take up roles previously unimagined i.e. directly controlling data and systems, outside the domain of the classical data center. With PCs and 4GL technology the notion dawned that more could be done with data than just servicing high-performance online transaction processing i.e. MIS (Management Information Systems) could be developed to run individual database applications for managerial decis ion making i.e. forefathers of today's DSS. Previously, data and IT were used exclusively to direct detailed operational decisions. The combination of PC and 4GL introduced the notion of a new paradigm i.e. a single database that could serve both operational high performance transaction processing and (limited) DSS, analytical processing, all at the same time.

The extract program

Shortly after the advent of massive online high-performance transactions, an innocent looking program called "extract" processing, began to show up.

The extract program was the simplest of all programs of its time. It scanned a file or database, used some criteria for selection, and, upon finding qualified data, transported the data into another file or database. Soon the extract program became very attractive, and flooded the information processing environment.

The spider web

Figure 1.2 shows that a "spider web" of extract processing programs began to form. First, there were extracts. Then there were extracts of extracts, then extracts of extracts of extracts, and it went on. It was common for large companies to be doing tens of thousands of extracts per day.

This pattern of extract processing across the organization soon became a routine activity, and even a name was coined for it. Extract processing gone out of control produced what was called the "naturally evolving architecture". Such architectures occurred when an organization had a relaxed approach to handling the whole process of hardware and software architecture. The larger and more mature the organization; the worse was the problems of the naturally evolving architecture.

Taken jointly, the extract programs or naturally evolving systems formed a spider web, also called "legacy systems" architecture.

<u>Crisis of Credi bility</u>

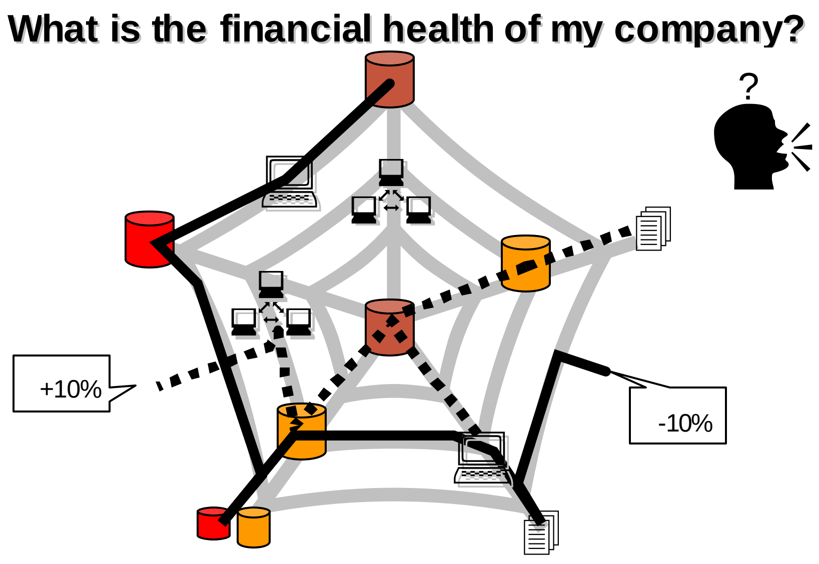


Figure-1.3: Crisis of Credibility: Who is right?

Consider the CEO of an organization who is interested in the financial health of his company. He asks the relevant departments to work on it and present the results. The organization is maintaining different legacy systems, employs different extract programs and uses different external data sources. As a consequence, Department-A which uses a different set of data sources, external reports etc. as compared to Department-B (as shown in Figure-1.3) comes with a different answer (say) sales up by 10%, as compared to the Department-B i.e. sales down by 10%. Because Department-B used another set of operational systems, data bases and external data sources. When CEO receives the two reports, he does not know what to do. CEO is faced with the option of making decisions based on politics and personalities i.e. very subjective and non-scientific. This is a typical example of the crisis in credibility in the naturally evolving archit ecture. The question is which group is right? Going with either of the findings could spell disaster, if the finding turns about to be incorrect. Hence the second important question, result of which group is credible? This is very hard to judge, since neit her had malicious intensions but both got a different view of the business using different sources.

Lecture Handout

Data Warehousing

Lecture No. 02

Why a DWH?

- Data recording and storage is growing.
- History is excellent predictor of the future.
- Gives total view of the organization.
- Intelligent decision-support is required for decision-making.

Data recording and storage is growing.

Moore's law on increase in performance of CPUs and decrease in cost has been surpassed by the increase in storage space and decrease in cost. Meaning, it is true that the cost of CPUs is going down and the performance is going up, but this is applicable at a higher rate to storage space and cost i.e. more and more cheap storage space is becoming available as compared to fast CPUs.

As you would have experienced, when you (or your father's) briefcase seems to be small as compared to the contents carried in it, it seems a good idea to buy a new and larger briefcase. However, after sometime the new briefcase too seems to be small for the contents carried. On the practical side, it has been noted that the amount of data recorded in an organization doubles every year and this is an exponential increase.

Reason-1: Data Sets are growing

How Much Data is that?		
1 MB	2^{20} or 10^{6} bytes	Small novel – 31/2 Disk
1 GB	2^{30} or 10^{9} bytes	Paper rims that could fill the back of a pickup van
1 TB	2^{40} or 10^{12} bytes	50,000 trees chopped and converted into paper and printed
2 PB	1 PB = 2^{50} or 10^{15} bytes	Academic research libraries across the U.S.
5 EB	1 EB = 2^{60} or 10^{18} bytes	All words ever spoken by human beings

Table-2.1: Quantifying size of data

- Size of Data Sets are going up ↑.

- Cost of data storage is coming down ↓.
  - Total hardware and software cost to store and manage 1 Mbyte of data
    - 1990: ~ $15
    - 2002: ~ ¢15 (Down 100 times)
    - By 2007: < ¢1 (Down 150 times)
  - A Few Examples
    - WalMart: 24 TB (Tera Byte)
    - France Telecom: ~ 100 TB
    - CERN: Up to 20 PB by 2006 (Peta Byte)
    - Stanford Linear Accelerator Center (SLAC): 500TB

A Ware House of Data
is NOT a
Data Warehouse

Someone says I have a data set of size 1 GB so I have a DWH can you beat this?
Someone else says, I have a data set of size 100 GB, can you beat this?
Someone else says, I have a 1 TB data set, who can beat this?

Who has a data warehouse? Not enough information, it is much more than just the size, it is a whole concept, it is NOT a shrink wrapped solution, it evolves. A company may have a TB of data and not have a data warehouse; while on the other hand, a company may have 500 GB of data and have a fully functional data warehouse.

Size is NOT
Everything

History is excellent predictor of the future
Secondly as I mentioned earlier the data warehouse has the historical data. And one thing that we have learned by using information is that, "past is the best predictor of the future". You use historical data, because it gives you an insight into how the environment is changing. Also you must have heard that "history repeats itself", however this repetition of history is not likely to be constant for all businesses or all events. Note that you just can't use the historical data to predict the future; you have to have to bring your own insight and experience to interpret how the environment is changing in order to predictthe future accurately and meaningfully.

Gives total view of the organization
So why would you want data warehouse in your organization? First of all a data warehouse gives a total view of an organization. If you look at the operational system i.e. the databases in most environments, the databases are designed around different lines of business. Consider the case of a Bank; a bank will typically have current accounts and savings accounts, foreign currency account etc. The bank will have an MIS system for leasing, and another system for managing credit cards and another system for every different kind of business they are in. However, nowhere they have the total view of the environment from the customer's perspective. The reason being, transaction processing systems are typically designed around functional areas, within a business environment. For good decision making you should be able to integrate the data across

the organization so as to cross the LoB (Line of Business). So the idea here is to give the total view of the organization especially from a customer's perspective within the data warehouse, as shown in Figure-2.1

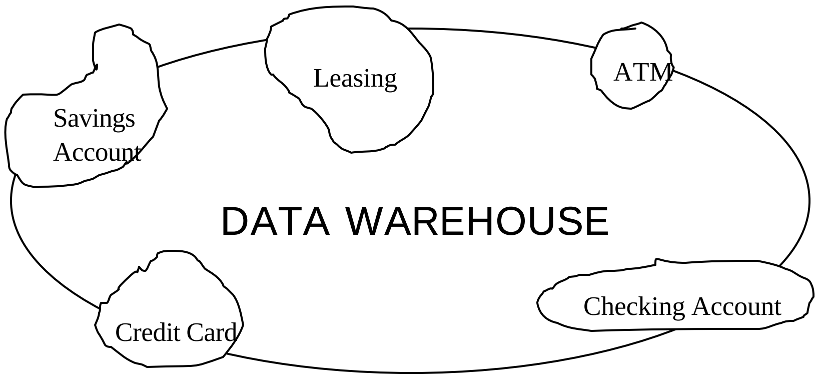


Figure-2.1: A Data Warehouse crosses the LoB

Intelligent decision-support is required for decision-making
Consider a bank which is losing customers, for reasons not known. However, one thing is for sure that the bank is losing business because of lost customers. Therefore, it is important, actually critical to understand which customers have left and why they have left. This will give you the ability to predict going forward (in time), to identify which customers will leave you (i.e. the bank). We are going to talk about this in the course using data mining algorithms, like clustering, classification, regression analysis etc. However, this being another example of using historical data to predict the future. So I can predict today, which customers will leave me in the next 3 months before they even leave. There can be, and there are whole courses on data mining, but we will just have an applied overview of data mining in this course.

Reason-2: Businesses demand intelligence

- Complex questions from integrated data.
- "Intelligent Enterprise"

DBMS Approach	Intelligent Enterprise
List of all items that were sold last month? List of all items purchased by Khizar? The total sales of the last month grouped by branch? How many sales transactions occurred during the month of January?	Which items sell together? Which items to stock? Where and how to place the items? What discounts to offer? How best to target customers to increase sales at a branch? Which customers are most likely to respond to my next promotional campaign, and why?

Table-2.2: Comparison of queries

Let's take a close look at the typical queries for a DBMS. They are either about listing the contents of tables or running aggregates of values i.e. rather simple and straightforward queries and fairly easy to program. The queries follow rather pre-defined paths into the database and are unlikely to come up with something new or abnormal.

Reason-3: Businesses want much more…

1. What happened?
2. Why it happened?
3. What will happen?
4. What is happening?
5. What do you want to happen?

These questions primarily point to what is called as the different stages of a Data Warehouse i.e. starting from the first stage, and going all the way to stage 5. The first stage is not actually a data warehouse, but a pure batch processing system. Note that as the stages evolve the amount of batching processing decreases, this being maximum in the first stage and minimum in the last or 5th stage. At the same time the amount of ad-hoc query processing increases. Finally in the most developed stage there is a high level of event based triggering. As the system moves from stage-1 to stage-5 it becomes what is called as an active data warehouse.

<u>What is a DWH?</u>

A <u>complete repository</u> of <u>historical</u> corporate data extracted from <u>transaction systems</u> that is available for ad-hoc access by knowledge workers

The other key points in this standard definition that I have also underlined and listed below are:

Complete repository

- All the data is present from all the branches/outlets of the business.
- Even the archived data may be brought online.
- Data from arcane and old systems is also brought online.

Transaction System

- Management Information System (MIS)
- Could be typed sheets (NOT transaction system)

Ad-Hoc access

- Does not have a certain predefined database access pattern.
- Queries not known in advance.
- Difficult to write SQL in advance.

Knowledge workers

- Typically NOT IT literate (Executives, Analysts, Managers).
- NOT clerical workers.
- Decision makers.

The users of data warehouse are knowledge workers in other words they are decision makers in the organization. They are not the clerical people entering the data or overseeing the transactions etc or doing programming or performing system design/analysis. These are really decision makers in the organization like General Manager Marketing, or Executive Director or CEO (Chief Operating Officer). Typically those decision makers are people in areas like marketing, finance and strategic planning etc.

Completeness: There is a misnomer here, about completeness. As per the standard definition a data warehouse is a complete repository of corporate data. The reality is that it can never be complete. We will discuss this in detail very shortly.

Transaction System: Unlike databases where data is directly entered, the input to the data warehouse can come from OLTP or transactional systems or other third party databases. This is not a rule, the data could come from typed or even hand filled sheets, as was the case for the census data warehouse.

Ad-Hoc access: It dose not have a certain repeatable pattern and it's not known in advance. Consider financial transactions like a bank deposit, you know exactly what records will be inserted deleted or updated. That's in OLTP system and in ERP system. But in a data warehouse there are really no fixed patterns. Say the marketing person, just sits down and thinks abou t what questions he/she has about customers and there behaviors and so on and they are typically using some tool to generate SQL dynamically and then that SQL gets executed and that you don't know in advance.

Although there may be some patterns of queries, but they are really not very predictable and the query patterns may change over time. Hence there are no predefined access paths into the database. That's why relational databases are so important for the data warehouse, because relational databases allow you to navigate the data in any direction that is appropriate using the primary, foreign key structure within the data model. Meaning, using a data warehouse, does not implies that we just forget about databases.

<u>**Another view of a DWH**</u>

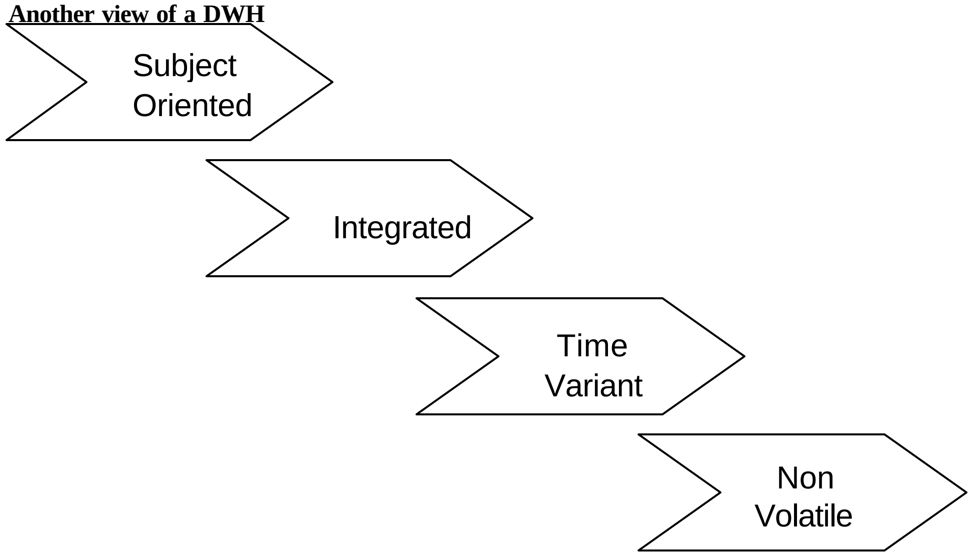


Figure-2.2: Another view of a Data Warehouse

Subject oriented. The goal of data in the data warehouse is to improve decision making, planning, and control of the major subjects of enterprises such as customer, products, regions, in contrast to OLTP applications that are organized around the work-flows of the company.

Integrated. The data in the data warehouse is loaded from different sources that store the data in different formats and focus on different aspects of the subject. The data has to be checked, cleansed and transformed into a unified format to allow easy and fast access.

Time variant. Time variant records are records that are created as of some moment in time. Every record in the data warehouse has some form of time variancy associated with it. In an OLTP system, the contents change with time i.e. updated such as bank account balance or mobile phone balance, but in a warehouse as the data is loaded; the moment usually becomes its time stamp.

Non-volatile. Unlike OLTP systems, after inserting data in the data warehouse it is neither changed nor removed. The only exceptions are when *false* or incorrect data gets inserted erroneously or the capacity of the data warehouse exceeded and archiving becomes necessary.

Lecture Handout

Data Warehousing

Lecture No. 03

It is a blend of many technologies, the basic concept being:

- Take all data from different operational systems.
- If necessary, add relevant data from industry.
- Transform all data and bring into a uniform format.
- Integrate all data as a single entity.
- Store data in a format supporting easy access for decision support.
- Create performance enhancing indices.
- Implement performance enhancement joins.
- Run ad-hoc queries with low selectivity.

A Data Warehouse is not something shrink-wrapped i.e. you take a set of CDs and install into a box and soon you have a Data Warehouse up and running. A Data Warehouse evolves over time, you don't buy it. Basically it is about taking/collecting data from different heterogeneous sources. Heterogeneous means not only the operating system is different but so is the underlying file format, different databases, and even with same database systems different representations for the same entity. This could be anything from different columns names to different data types for the same entity.

Companies collect and record their own operational data, but at the same time they also use reference data obtained from external sources such as codes, prices etc. This is not the only external data, but customer lists with their contact information are also obtained from external sources. Therefore, all this external data is also added to the data warehouse.

As mentioned earlier, even the data collected and obtained from within the company is not standard for a host of different reasons. For example, different operational systems being used in the company were developed by different vendors over a period of time, and there is no or minimal evenness in data representation etc. When that is the state of affairs (and is normal) within a company, then there is no control on the quality of data obtained from external sources. Hence all the data has to be transformed into a uniform format, standardized and integrated before it c an go into the data warehouse.

In a decision support environment, the end user i.e. the decision maker is interested in the big picture. Typical DSS queries do not involve using a primary key or asking questions about a particular customer or account. DSS queries deal with number of variables spanning across number of tables (i.e. join operations) and looking at lots of historical data. As a result large number of records are processed and retrieved. For such a case, specialized or different database architectures/topologies are required, such as the star schema. We will cover this in detail in the relevant lecture.

Recall that a B-Tree is a data structure that supports dictionary operations. In the context of a database, a B-Tree is used as an index that provides access to records without actually scanning the entire table. However, for very large databases the corresponding B-Trees becomes very large. Typically the node of a B-Tree is stored in a memory block, and traversing a B-Tree involves O(log n) page faults. This is highly undesirable, because by default the height of the B-Tree would be very large for very large data bases. Therefore, new and unique indexing techniques are required in the DWH or DSS environment, such as bitmapped indexes or cluster index etc. In some cases the designers want such powerful indexing techniques, that the queries are satisfied from the indexes without going to the fact tables.

In typical OLTP environments, the size of tables are relatively small, and the rows of interest are also very small, as the queries are confined to specifics. Hence traditional joins such as nested-loop join of quadratic time complexity does not hurt the performance i.e. time to get the answer. However, for very large databases when the table sizes are in millions and the rows of interest are also in hundreds of thousands, nested-loop join becomes a bottle neck and is hardly used. Therefore, new and unique forms of joins are required such as sort-merge join or hash based join etc.

Run Ad-Hoc queries with low <u>Selectivity</u>
Have already explained what is meant by ad-hoc queries. A little bit about selectivity is in order. Selectivity is the ratio between the number of unique values in a column divided by the total number of values in that column. For example the selectivity of the gender column is 50%, assuming gender of all customers is known. If there are N records in a table, then the selectivity of the primary key column is 1/N. Note that a query consists of retrieving records based on a a combination of different columns, hence the choice of columns determine the selectivity of the query i.e. the number of records retrieved divided by the total number of records present in the table.

In an OLTP (On-Line Transaction Processing) or MIS (Management Information System) environment, the queries are typically Primary Key (PK) based, hence the number of records retrieved is not more than a hundred rows. Hence the selectivity is very high. For a Data Warehouse (DWH) environment, we are interested in the big picture and have queries that are not very specific in nature and hardly use a PK. As a result hundreds of thousands of records (or rows) are retrieved from very large tables. Thus the ratio of records retrieved to the total number of records present is high, and hence the selectivity is low.

How is it different?

Decision making is Ad-Hoc

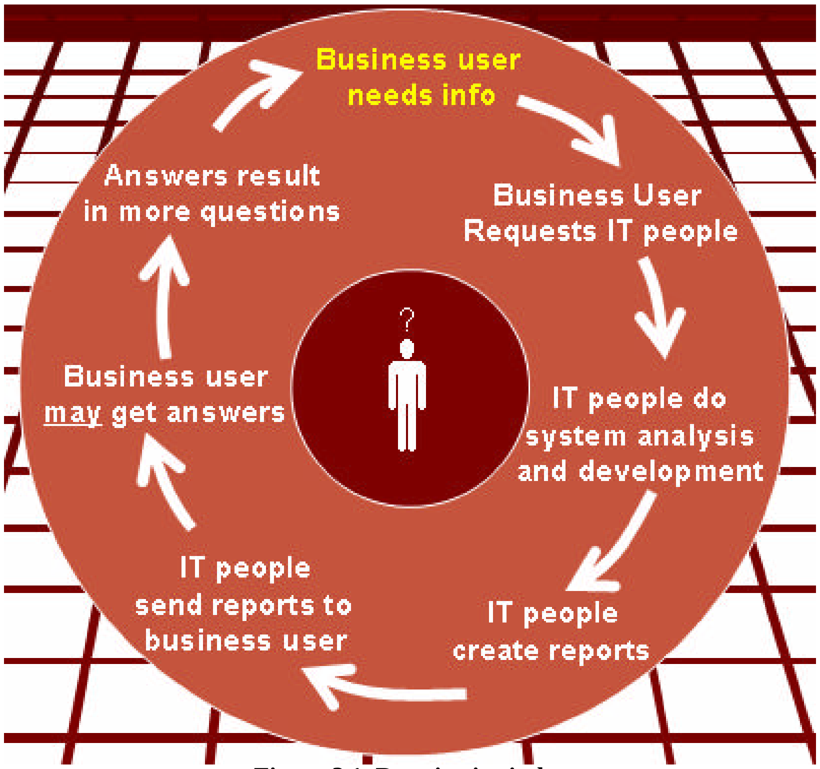

Business user needs info
Business User Requests IT people
IT people do system analysis and development
IT people create reports
IT people send reports to business user
Business user may get answers
Answers result in more questions
?

Figure-3.1: Running in circles

Consider a decision maker or a business user who wants some of his questions to be answered. He/she sets a meeting with the IT people, and explains the requirements. The IT people go over the cycle of system analysis and design, that takes anywhere from couple of weeks to couple of months and they finally design and develop the system. Happy and proud with their achievement the IT people go to the business user with the reporting system or MIS system. After a learning curve the business users spends some time with the brand new system, and may get some answers to the required questions. But then those answers results in more questions. The business user has no choice to meet the IT people with a new set of requirements. The business user is frustrated that his questions are not getting answered, while the IT people are frustrated that the business user always changes the requirements. Both are correct in their frustration.

<ins>Different patterns of hardware utilization</ins>

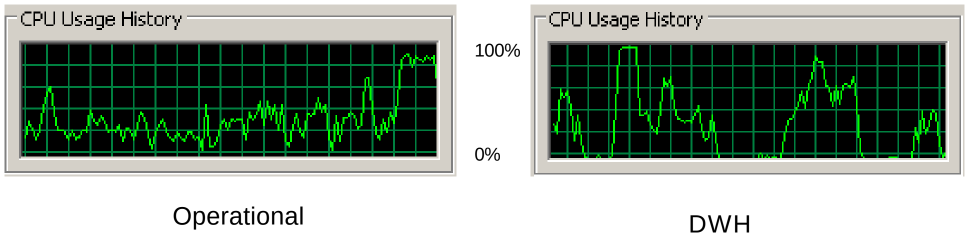


Figure-3.2: Different patterns of CPU Usage

Although there are peaks and valleys in the operational processing, but ultimately there is relativ ely static pattern of utilization. There is an essentially different pattern of hardware utilization in the data warehouse environment i.e. a binary pattern of utilization, either the hardware is utilized fully or not at all. Calculating a mean utilization for a DWH is not a meaningful activity. Therefore, trying to mix the two environments is a recipe for disaster. You can optimize the machine for the performance of one type of application, not for both.

Bus vs. Train Analogy

Consider the analogy of a bus and train. I believe you can find dozens of buses operating between Lahore and Rawalpindi almost every 30 minutes. As a consequence, literally there are buses moving between Lahore and Rawalpindi almost continuously through out the day. But how many times a dedicated train moves between the two cities? Only twice a day and carries a bulk of passengers and cargo. Binary operation i.e. either traveling or not. The train can NOT be optimized for every 30-min travel, it will never fill to capacity and run into loss. A bus can not be optimized to travel only twice, it will stand idle and passengers would take vans etc. Bottom line: Two modes of transportation, can not be interchanged.

Combines historical & Operational Data

- Don't do data entry into a DWH, OLTP or ERP are the source systems.
- OLTP systems don't keep history, cant get balance statement more than a year old.
- DWH keep historical data, even of bygone customers. Why?
- In the context of bank, want to know why the customer left?
- What were the events that led to his/her leaving? Why?
- Customer retention.

Why keep historical data?

The data warehouse is different because, again it's not a database you do data entry. You are actually collecting data from the operational systems and loading into the DWH. So the transactional processing systems like the ERP system are the source systems for the data warehouse. You feed the data into the data warehouse. And the data warehouse typically collects data over a period of time. So if you look at your transactional processing OLTP systems, normally such systems don't keep very much history. Normally if a customer leaves or expired, the OLTP systems typically purge the data associated with that customer and all the transactions off the database after some amount of time. So normally once a year most business will purge the database of all the old customers and old transactions. In the data warehouse we save the historical data. Because you don't need historical data to do business today, but you do need the historical data to understand patterns of business usage to do business tomorrow, such why a customer left?

How much History?

- Depends on:

    - Industry.
    - Cost of storing historical data.
    - Economic value of historical data.

- Industries and history
    - **Telecomm** calls are much much more as compared to bank transactions - 18 months of historical data.
    - **Retailers** interested in analyzing yearly seasonal patterns - 65 weeks of historical data.
    - **Insurance** companies want to do actuary analysis, use the historical data in order to predict risk- 7 years of historical data.

Hence, a DWH NOT a <u>complete repository</u> of data

How back do you look historically? It really depends a lot on the industry. Typically it's an economic equation. How far back depends on how much dose it cost to store that extra years work of data and what is it's economic value? So for example in financial organizations, they typically store at least 3 years of data going backward. Again it's typical. It's not a hard and fast rule.

In a telecommunications company, for example, typically around 18 months of data is stored. Because there are a lot more call details records then there are deposits and withdrawals from a bank account so the storage period is less, as one can not afford to store as much of it typically. Another important point is, the further back in history you store the data, the less value it has normally. Most of the times, most of the access into the data is within that last 3 months to 6 months. That's the most predictive data.

In retail business, retailers typically store at least 65 weeks of data. Why do they do that? Because they want to be able to look at this season's selling history to last season's selling history. For example, if it is Eid buying season, I want to look at the transit-buying this Eid and compare it with the year ago. Which means I need 65 weeks in order to get year going back, actually more then a year. It's a year and a season. So 13 weeks are additionally added to do the analysis. So it really depends a lot on the industry. But normally you expect at least 13 months.

Economic <u>value</u> of data
Vs.
Storage <u>cost</u>

Data Warehouse a
complete repository of data?

This raises an interesting question, do we decide about storage of historical data using only time, or consider space also, or both?

Usually (but not always) periodic or batch updates rather than real-time.

- The boundary is blurring for active data warehousing.
- For an ATM, if update not in real-time, then lot of real trouble.
- DWH is for strategic decision making based on historical data. Wont hurt if transactions of last one hour/day are absent.
- Rate of update depends on:
    - Volume of data,
    - Nature of business,
    - Cost of keeping historical data,
    - Benefit of keeping historical data.

It's also true that in the traditional data warehouse the data acquisition is done on periodic or batch based, rather then in real time. So think again about ATM system, when I put my ATM card and make a withdrawal, the transactions are happening in real time, because if they don't the bank can get into trouble. Someone can withdraw more money then they had in their account! Obviously that is not acceptable. So in an online transaction processing (OLTP) system, the records are updated, deleted and inserted in real-time as the business events take place, as the data entry takes place, as the point of sales system at a super market captures the sales data and inserts into the database.

In a traditional data warehouse that is not true. Because the traditional data warehouse is for strategic decision-making not for running day to day business. And for strategic decision making, I don't need to know the last hours worth of ATM deposits. Because strategic decisions take the long term perspective. For this reason, and for efficiency reasons normally what happens is that in the data warehouse you update on some predefined schedule basis. May be it's once a month, maybe it's once a weak, maybe it's even once every night. It depends on the volume of data you are working with, and how important the timings of the data are and so on.

Deviation from the PURIST approach

Let me first explain what/who a purist is. A purist is an idealist or traditionalist who wants everything to be done by the book or the old arcane ways (only he/she knows), in short he/she is not pragmatic or realist. Because the purist wants everything perfect, so he/she has good excuses of doing nothing, as it is not a perfect world. When automobiles were first invented, it was the purists who said that the automobiles will fail, as they scare the horses. As Iqbal very rightly said "*Aina no sa durna Tarzay Kuhan Pay Arna…*"

As data warehouses become mainstream and the corresponding technology also becomes mainstream technology, some traditional attributes are being deviated in order to meet the increasing demands of the user's. We have already discussed and reconciled with the fact that a data warehouse is NOT the complete repository of data. The other most noticeable deviations being time variance and nonvolatility.

Deviation from Time Variance and Nonvolatility

As the size of data warehouse grows over time (e.g., in terabytes), reloading and appending data can become a very tedious and time consuming task. Furthermore, as business users get the "hang of it" they start demanding that more up-to-date data be available in the data warehouse. Therefore, instead of sticking to the traditional data warehouse characteristic of keeping the data nonvolatile and time variant, new data is being added to the data warehouse on a daily basis, if not on a real-time basis and at the same time historical data removed to make room for the "fresh" data. Thus, new approaches are being made to tackle this task. Two possible methods are as follows:

- Perform hourly/daily batch updates from shadow tables or log files. Transformation rules are executed during the loading process. Thus, when the data reaches the target data warehouse database, it is already transformed, cleansed and summarized.

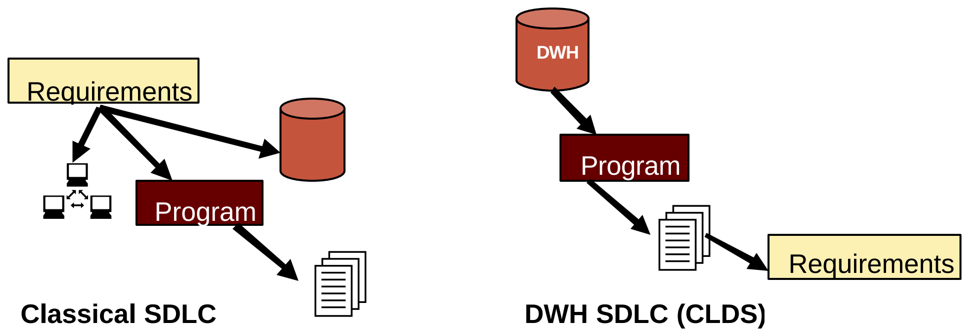
Perform real-time updates from shadow tables or log files. Again, transformation rules are executed during the loading process. Instead of batch updates, this takes place on a per transaction basis that meets certain business selection criteria.

<u>Starts with a 6x12 availability requirement ... but 7x24 usually becomes the goal.</u>

- Decision makers typically don’t work 24 hrs a day and 7 days a week. An ATM system does.
- Once decision makers’ start using the DWH, and start reaping the benefits, they start liking it…
- Start using the DWH more often, till want it available 100% of the time.
- For business across the globe, 50% of the world may be sleeping at any one time, but the businesses are up 100% of the time.
- 100% availability not a trivial task, need to take into loading strategies, refresh rates etc.

If we look at the availability requirements for a data warehouse, normally you start out with 6 days a week, 12 hours a day. The decision makers are not going to come during the middle of a weekend to ask marketing questions. They don’t normally do that. They don’t come in midnight to ask these questions either. However, a transaction processing system, like the ATM systems, should be available all the time. I can go to an ATM at midnight if I want to, and withdraw money. That’s not usually the requirement in the beginning of the data warehouse project, but as the data warehouse matures, availability becomes much more important. Because the decision makers start accessin g and using the data more often. They start doing maybe interactive kind of analysis in the day and data mining kind of things over night. We will talk more about data mining later. So you can start out with 6× 12 availability but will usually evolve in to 7×24 over a period of time. So one of the things we are going to talk about, although there will not be a major focus is about the tradeoffs that you make in availability. How do I design my data warehouse for 100% availability? So that it is always available for quires. And it turns out that there are very subtle interactions between availability and data loading. How do I make sure the data is available for querying while uploading the data? These issues turn out to have some tricky subtleties in that kind of environment.

Does not follows the traditional development model



Classical SDLC

- Requirements gathering
- Analysis
- Design
- Programming
- Testing
- Integration
- Implementation

DWH SDLC (CLDS)

- Implement warehouse
- Integrate data
- Test for biasness
- Program w.r.t data
- Design DSS system
- Analyze results
- Understand requirement

Figure-4.1: Comparison of SDLC & CLDS

Operational environment is created using the classical systems development life cycle. The DWH on the other hand operates under a very different life cycle. Sometimes called CLDS i.e. the reverse of SDLC. The classical SDLC is requirement driven. In order to build the system, you must first understand the end user or business user requirements. Then you go into the stages of design and development typically taught in software engineering. The CLDS is almost exactly the reverse of SDLC. The CLDS starts with the data. Once the data is in hand, it is integrated, and then tested to see what bias there is, if any. Programs are then written against the data. The results are analyzed, and finally the requirements of the system are understood. Some Data Warehouse developers/vendors overuse this "natural" approach and add extensive charges for adding features or enhancing features already present once the end users get the "hang" of the system. The CLDS is a classical data driven development life cycle. Trying to apply inappropriate tools and techniques of development will only result in waste of effort/resources and confusion

Typical queries

OLTP (On Line Transaction Processing) specific query

```
Select tx_date, balance from tx_table
Where account_ID = 23876;
```

DWH specific query

```
Select balance, age, sal, gender from customer_table and tx_table
Where age between (30 and 40) and
Education = 'graduate' and
CustID.customer_table = Customer_ID.tx_table;
```

Lets take a brief look a the two queries, the results of comparing them are summarized in table 4.1 below:

OLTP	DWH
Primary key used	Primary key NOT used
No concept of Primary Index	Primary index used
May use a single table	Uses multiple tables
Few rows returned	Many rows returned
High selectivity of query	Low selectivity of query
Indexing on primary key (unique)	Indexing on primary index (non-unique)

Table-4.1: Comparison of OLTP and DWH for given queries

	Data Warehouse	OLTP
Scope	* Application –Neutral * Single source of "truth" * Evolves over time * How to improve business	* Application specific * Multiple databases with repetition * Off the shelf application * Runs the business
Data Perspective	* Historical, detailed data * Some summary * Lightly denormalized	* Operational data * No summary * Fully normalized
Queries	* Hardly uses PK * Number of results returned in thousands	* Based on PK * Number of results returned in hundreds
Time factor	* Minutes to hours * Typical availability 6x12	* Sub seconds to seconds * Typical availability 24x7

Table-4.2: Detailed comparison of OLTP and DWH

Comparisons of response times

- On-line analytical processing (OLAP) queries must be executed in a small number of seconds.
    - Often requires denormalization and/or sampling.
- Complex query scripts and large list selections can generally be executed in a small number of minutes.

- Sophisticated clustering algorithms (e.g., data mining) can generally be executed in a small number of hours (even for hundreds of thousands of customers).

So there is one class of decision support environment called OLAP (Online Analytical Processing). The idea is that I am doing iterative decision making, using point and clicking, drilling down into data, and refining my decision model and I should be able to "execute" queries in an OLAP environment in a small number of seconds. How is it possible to execute queries in a small numbers of seconds? And potentially working with billions of rows of data? I have to be a bit clever here. I have to consider special techniques like sampling, like de-normalization, special indexing techniques and other smart kinds of techniques. We will talk about those techniques when we go to the relevant parts. And how do I design a database which has these characteristics?

If I have got complex combination of tables and a large list selection and if it takes small number of minutes that's acceptable. If I got a lot of iterative analysis to do by pulling a lis t of names and customers and some information about them to send a direct mail to those customers, then it is acceptable to assume that that such queries don't have to be executed in few seconds. Of course as long as I am going to find them, it's all right.

Putting the pieces together

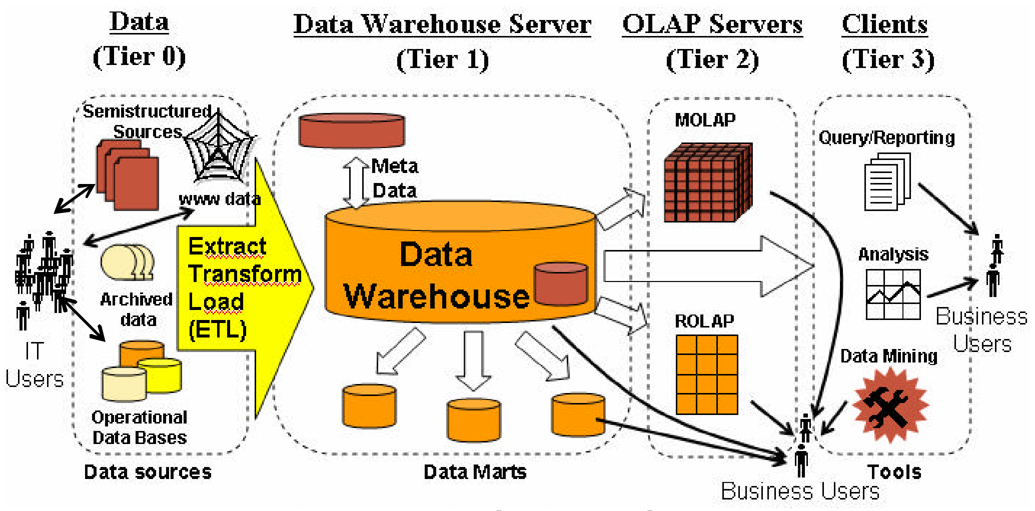


Figure-4.2: Putting the pieces together

Figure 4.2 gives a complete picture of how different "pieces" of a data warehouse fit together. We start with extracting data from different sources, transform and clean that data and then load it into the data warehouse. The data warehouse itself has to have specialized schemas to give quick answers to typical macroscopic queries. Data marts are created out of the data warehouse to service the needs of different departments such as marketing, sales etc such that they don't have to work with the heavy load of the complete data warehouse. Once the data warehouse is in place (as briefly discussed before) data cubes are created for OLAP based drill-down "all" possible queries. And we move further and use smart tools such as data mining clustering and classification techniques.

Why this is hard?

- Data sources are unstructured & heterogeneous.
- Requirements are always changing.
- Most computer scientist trained on OTLP systems, those concepts not valid for VLDB & DSS.
- The scale factor in VLDB implementations is difficult to comprehend.
- Performance impacts are often non-linear O(n) Vs. O(nlog_n) e.g. scanning vs. indexing.
- Complex computer/database architectures.
- Rapidly changing product characteristics.
- And so on...

In early days of data warehousing you would never even conceive giving business end user access to the data. They would always have to go through a data center, some kind of reporting or MIS system, that's really not true any more. The hard part of the problem now is architecting, designing and building these systems. And it is particularly hard at the high end. Because in a data warehouse environment, there are no stable requirements-change is constant. This is one of the main reasons why SDLC system dose not work. By the time you collect all the requirements following the SDLC approach and start designing the system the requirements may have completely changed. They are completely meaningless. Because any question that was interesting 6 months ago is definitely not interesting today.

The other issue is with very large databases scale dose strange things in the environment. And this is very important for computer scientist to understand because at small scale (i.e. hundreds of rows) an O(n log n) algorithm and an $O(n^2)$ algorithms are not much different in terms of performance. It dose not matter if you are using bubble sort or a heap sort for small amount of data; nobody cares as the difference in performance is not noticeable. However, when you get millions of records or a billion records, then it starts to hurt. So when you get very large data bases, and manipulating large amount of data because we are retaining history, this becomes a big problem and a proper design is required to be in place. Else the scale will kill you. The difference between O(n log n) and an $O(n^2)$ is huge when you get to billions of records.

High level implementation steps

Phase-I
1. Determine Users' Needs
2. Determine DBMS Server Platform
3. Determine Hardware Platform
4. Information & Data Modeling
5. Construct Metadata Repository

Phase-II
6. Data Acquisition & Cleansing
7. Data Transform, Transport & Populate

8. Determine Middleware Connectivity
9. Prototyping, Querying & Reporting
10. Data Mining
11. On Line Analytical Processing (OLAP)

Phase-III
12. Deployment & System Management

If you look at the high level implementations steps for a data warehouse it is important that you are driving the design of the data warehouse in the context of the data warehouse by the business requirements, and not driven by the technology.

So you start with the business requirements and say ok what problems I am trying to solve here. Am I going to do fraud detection or do customer retention analysis? What are the specifications of the problems that we have discussed? And identify the key source of these problems so you can understand what is going to be the cost and time required to fix them. But make sure this is done in the context of a logical data model that expresses the underlying business relationship of the data.

The data warehouse should be able to support multiple applications, multiple workloads against a single source of truth for the organization. So by identifying the business opportunity, you identify the information required, and then in the next stage, how to schedule those deliverables that are most important to the business. Ask yourself, what can I do in a 90 days time period to deliver values to the business? And then based on what I have decided here do the software and hardware selection. Remember that software is hardware, and hardware is easy ware. Because the software is much more complex to scale (algorithm complexity) then the hardware. Hardware is getting cheaper over time. So you drive typically from the software not the hardware.

Lecture Handout

Data Warehousing

Lecture No. 05

Types of Data Warehouses

- Financial
- Telecommunication
- Insurance
- Human Resource

Financial DWH

- First data warehouse that an organization builds. This is appealing because:
    - Nerve center, easy to get attention.
    - In most organizations, smallest data set.
    - Touches all aspects of an organization, with a common denomination i.e. money.
    - Inherent structure of data directly influenced by the day-to-day activities of financial processing.
    - Financial data warehouses suffer from an anomaly of inability to match balanced accounts, due to many legitimate reasons.

Financial data warehouses are often the first data warehouse that an organization builds. This is appealing because:

Financial data is ALWAYS at the nerve center of the organization. Therefore, getting attention drawn to a well-built financial data warehouse is a very easy thing to do. In most organizations (but not all), financial data represents one of the smallest volumes of data that exist. Finance cuts across all of the aspects of the corporate data and has a common denominator i.e. money. Financial data inherently has a structure of data directly influenced by the day-to-day activities of financial processing and the list goes on. For these reasons, finance becomes a very good target for beginning to build the corporate data warehouse.

Financial data warehouses have some severe drawbacks that are not found elsewhere. When purists use a financial data warehouse, they are overjoyed to find that it is almost impossible to reconcile down to the rupee. Again and again they will report that they saw a report yesterday that stated the balance to be Rs. 12,345,678 but when did the same report today the using our accounting MIS system the balance was Rs. 13,245,678. Hence one can not trust the new system☺.

Note that it is unlikely that the financial data warehouse would balance to the last rupee, such a notion is completely wrong for a number of reasons. The accounting periods may be different in different operational systems, but in a data warehouse, it is the corporate calendar. The classifications of regions may change. In the operational system Northern Pakistan may consist of Murree, Abbotabad etc. but in the national data warehouse, Northern Pakistan consists of every major city north of Lahore. In the operational system (for example) sale or oranges is recorded in dozen in one part of Pakistan, while in other they are sold by weight and in another part by crates. However, in the national data warehouse, they are accounted for based on dozen and the list goes on.

Telecommunication

Dominated by sheer volume of data.

Many ways to accommodate call level detail:

- Only a few months of call level detail,
- Storing lots of call level detail scattered over different storage media,
- Storing only selective call level detail, etc.
- Unfortunately, for many kinds of processing, working at an aggregate level is simply not possible.

Telecommunications Data Warehouse
The telecommunications data warehouse is dominated by the sheer volume of data generated at the call level subject area. There are many other types of data in this environment as well. No other data warehouse environment is dominated by the size of one of the subject areas as call level detail dominates a telecommunications data warehouse.

There are many ways that call level detail can be accommodated:

- Only a few months of call level detail,
- Storing lots of call level detail scattered over different storage media,
- Summarizing or aggregating call level detail,
- Storing only selective call level detail, and so forth.

Unfortunately, for many kinds of processing, the only way it can be accommodated is by working at the call level detail. Working at a summary level or an aggregate level simply is not a possibility.

Insurance

Insurance data warehouses are similar to other data warehouses BUT with a few exceptions.

- Stored data that is very, very old, used for actuarial processing.

- Typical business may change dramatically over last 40-50 years, but not insurance. Therefore, if someone says that the insurance business has not changed much over the last 40-50 years would be telling the truth.

- In retailing or telecomm there are a few important dates, but in the insurance environment there are many dates of many kinds. In the retail sales environment there are a few dates such as: sale date, stocking date, and perhaps manufacturing date. In the banking environment there is the transaction date. However, in the insurance environment there are dates everywhere of every kind.

- Long operational business cycles, in years. Thus the operating speed is different. Consider a bank, you insert the ATM card and draw the money instantly, or deposit a check and it is cashed at most in a week. You go to store and purchase an item, and instantly make a telephone call. However, in an insurance environment a claim is made and it may take several years before it is settled.

- Transactions are not gathered and processed, but are in kind of “frozen” state.

- Thus a very unique approach of design & implementation.

Differences between different types of data warehouses

- Financial data warehouses, often the first a corporation builds. However, it will not reconcile down to the last rupee with the existing operational financial environment.

- Insurance data warehouses are similar to other data warehouses with a few exceptions: such as the length of time that insurance data warehouses exists, in terms of the dates found in the business, and in terms of the operational business cycle.

- The human resources data warehouse is different from others because this without doubt has only one major subject area (yes you guessed it right!).

- The telecommunications data warehouse is dominated by the sheer volume of data generated at one subject area.

Indeed there are many types of data warehouses, each with their own peculiarities and all of them can not be discussed here.

Typical Applications of DWH

There are, and there can be many many applications of a data warehouse. It is not possible to discuss all of them. Some representative applications are listed to be discussed as follows:

- Fraud detection.
- Profitability analysis.
- Direct mail/database marketing.
- Credit risk prediction.
- Customer retention modeling.
- Yield management.
- Inventory management.

ROI on any one of these applications can justify hardware/software and consultancy costs in most organizations.

Fraud detection

- By observing data usage patterns.
- People have typical purchase patterns.
- Deviation from patterns.
- Certain cities notorious for fraud.
- Certain items bought by stolen cards.
- Similar behavior for stolen phone cards.

You can look at patterns of data usage, and behavior of customer and detect fraud very easily. Lets suppose you are a Credit Card Company. You know that Danyal as a customer has certain behaviors. He tends to buy airplane tickets, make hotel reservations, rents cars using his credit card. When he goes shopping, it is usually only on a weekend and only on certain stores that he goes shopping and buys sports goods and action clothing. And then all of sudden you see very different behaviors coming in, such as credit card transactions in Karachi. Karachi by the way has very high stolen credit card rate. And the patterns of buying behavior are stereos, and CDs and TVs i.e. electronic goods that are typically the things that people buy on stolen credit cards. You can notice and say "look, this is not airplane or hotel reservations, this does not look like Danyal". It is not in the city where Danyal lives or goes to very often. Although he was there a week ago, so maybe this is how you can tell that when and where his credit card was stolen. And the purchases were made after he left the city. Hence "high probability of fraud".

So I can predict based on the behavior of the individuals, when things out of the typical routine happen. Then I can stop the fraud happening even before it has spread. That's the example in credit card but you can do the same thing with telephone cards. I am sure you have used these telephone cards. You punch in your number and you get access to telephone network. You find people at airports or railway stations who make there living by stealing the telephone card codes. They stay near the phones at the air port, they act like they are talking on the phone, but they are really looking over your shoulder when you are punching in the code. And then they memorize the code and then go sell it on the street to people who make long distance phone calls to all their friends in Brazil and Australia all these expensive places. You can look at the calling patterns and say look "Danyal makes regular calls to areas like say Rawalpindi and to friends in Lahore" and all of sudden we see all these phone calls to Toba Taik Singh or Bhawalpur, meaning there is some thing wrong here.

Profitability Analysis

- Banks know if they are profitable or not.
- Don't know which customers are profitable.
- Typically more than 50% are NOT profitable.
- Don't know which one?
- Balance is not enough, transactional behavior is the key.
- Restructure products and pricing strategies.
- Life-time profitability models (next 3-5 years).

It is another example. If you go to an average bank, to Pakistan or any where in the world, the bank will know if they are profitable or not. But they don't know which customers are profitable or not, and at most banks this is true, especially true for retail banks. These banks do business with the consumers, such that more then 50 % of the customers are unprofitable. In other word the bank is loosing money on 50% of their customers. But they don't know which 50%? That's the problem. So the idea behind profitability analysis is to do the analysis to figure out which customers are profitable and which customers are not profitable. And then based on that they can restructure their product offering and there pricing strategies to do more profitable business. I used a banking example but same is true for Telecommunications Company or any other consumer oriented business that requires access to detail data. It is not sufficient to just know the account balance, but also transactional behavior and so on to do profitability analysis. And once I know profitability retrospectively, I can also build predictive models to understand what it's going to be prospectively. So I can build what's called lifetime value models to using the historical data to predict what the future profitability will be for the next 3 or 5 years.

Direct mail marketing

- Targeted marketing.
- Offering high bandwidth package NOT to all users.
- Know from call detail records of web surfing.
- Saves marketing expense, saving pennies.
- Knowing your customers better.

Assume you are a telecommunications company, and you have an offering that you can make such as to give value added Internet access through mobile phones. You can send a letter or an SMS to every customer in the country and tell them about this value added Internet access service. But 95% of that mail will be completely wasted. Because a small percentage of customers will be interested in such as service. So the question here is how you your message to the right people in a meaningful way. If you have the call detailed records, you can tell very very easily which of your customer's access the Internet and which of them just make missed calls. Because the patters of the call detail records of Internet access are completely different from making voice calls and making missed calls. They have different durations; they have different data bandwidth requirement and so on. So by having access to detail data, you can target those people that are better to get a second account, or better to get a value added connection/service.

Credit risk prediction

- Who should get a loan?
- Customer segregation i.e. stable vs. rolling.
- Qualitative decision making NOT subjective.
- Different interest rates for different customers.
- Do not subsidize bad customer on the basis of good.

Credit risk prediction is another area where data warehouse is used quite frequently. Let's say I and my friend both apply for home loan to construct a house. How would the bank determine which one of those should get a loan? So typically they will say, ok I have been employed for 5 years, have good salary, married, have children etc etc and then based on the context of application then they decide yes we think you are a low risk we will give you the loan. Or say no you are a high risk because you had a job only for one year, your salary is not high, and you have lots of change of jobs recently, so you know we don't feel so good about the risk we are taking, so we won't give you the loan.

So typically today it is based on what is called qualitative decision-making. Basically in the offices these are just references say OK Saeed is a nice person or is related to me or is my neighbor so we give him the loan. And I don't really know Waleed over here so we don't give him a loan. This is not the best way to give fair loan to people and is not the best way to manage risk in the business. So with credit risk, the idea is to build quantitative models, which predict risk based on historical data. And not only I use the historical data to predict the risk, if I get more sophisticated, what I can do is called risk based pricing. So today lets say you offer me (Saeed) a loan, and also to the other applicant i.e. Waleed also loan, and both are given the loan at the same interest rates. However, if Waleed is a lower risk then Saeed then why should both pay the same interest rates? If Waleed is a low risk, then he should get a lower interest rate other wise the bank is subsidizing the higher risk loan. So what's happening is that you are using very precise credit prediction models in order to adjust the interest to give lower interest to people who are at lower risk of default. And again it should use some data mining algorithms to predict the risk, which is called the risk index on every individual consumer. So that every consumer gets a loan at an interest that is appropriate to their risk level.

Yield Management

- Works for fixed inventory businesses.
- The price of item suddenly goes to zero.
- Item prices vary for varying customers.
- Example: Air Lines, Hotels, Car rentals etc.
- Price of (say) Air Ticket depends on:
    - How much in advance ticket was bought?
    - How many vacant seats were present?
    - How profitable is the customer?
    - Ticket is one-way or return?

Yield management works, for example for sophisticated airlines which use a variable pricing mechanism for the seats. Say you are flying on a plane; 90% of the time the person sitting next to you pays a different price for the ticket as compared to yours. The seats are basically same, yet you pay a different price for the ticket, why is it that? Because the airline used a sophisticated method to see how much in advance the person bought the ticket, and how many seats were available on the plane at the time the person bought the ticket. Or did they buy a round ticket or one-way ticket; there are all kinds of sophisticated ways to do the pricing. Yield management helps decide what should be the price at any time for that airline seat.

Yield management is important for almost any type of fixed inventory business. For example, once the plane leaves, the price of the seat if it is empty or full means nothing any more. So the goal of the airline is to fill every seat with the maximum possible price for that seat. The same is true for hotels. For example a three star hotel had an empty room last night. If they can get Rs. 1500 for that night it is better then having nothing at all. So someone who booked the room for example as a group, would have paid lower price per room as compared to someone who is booking only one room. The price is different, because the hotels use yield management to fill out their hotel rooms as profitably as they can. This is applicable to car rentals also.

Agriculture Systems

- Agri and related data collected for decades.
- Metrological data consists of 50+ attributes.
- Decision making based on expert judgment.

- Lack of integration results in underutilization.
- What is required, in which amount and when?

Each year different government departments and agencies in Pakistan create tens of thousands of digital and non digital files from thousands of pest-scouting surveys, yield surveys, metrological data collection, river flows etc. This data collection has been going on for decades. The data collected has never been compiled, standardized and integrated to give a complete picture. Thus the lack of data integration and standardization contributes to an under-utilization of historical data, and inevitably results in an inability to perform any scientific predictive analysis for effective decision support and policy making. An Agriculture data warehouse is the answer, as processing 100+ variables by experts for large historical data is not possible. This has repeatedly resulted in tragic outcomes.

Although the benefits are many, but are not without some major issues or problems, some of them are as follows:

<u>Major Issues</u>

- Unavailability of data and in an illegible format and form.
- Most of the detail data is not digitized; therefore it can not be readily used by computers.
- Dispersion of data across multiple and geographically displaced organizations.
- Even though the said organizations are required to share the data with common citizens, no official data sharing mechanism is currently in place.

Until the requisite training is provided, government employees and farmers are not equipped with enough technical and literate skills to use latest information technology tools to use the data warehouse

Lecture Handout

Data Warehousing

Lecture No. 06

Normalization

Before we begin our discussion about the normal forms, it's important to understand that what we will discuss are guidelines and guidelines only i.e. not a dejure standard. Sometimes (especially in a DSS environment), it becomes necessary to drift from this purist approach to meet practical business requirements of performance. However, when deviation take place, it's extremely important to evaluate any possible consequences these deviations would cause and the corresponding inconsistencies and anomalies this is what de-normalization is about.

What is normalization?

What are the goals of normalization?

- Eliminate redundant data.
- Ensure data dependencies make sense.

What is the result of normalization?

What are the levels of normalization?

Always follow purist's approach of normalization?
NO

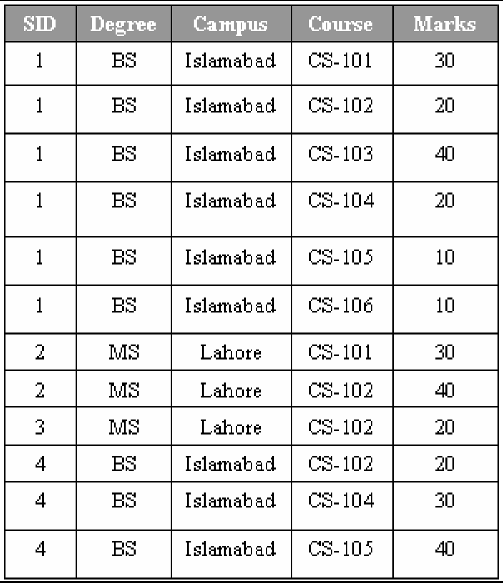
Normalization is the process of efficiently organizing data in a database by decomposing (splitting) a relational table into smaller tables by projection. There are basically two goals of normalization as follows:

1. Eliminate redundant data (for example, storing the same data in more than one table)

2. Ensuring data dependencies make sense (only storing related data in a table).

Both of these are worthy goals, as they reduce the amount of space a database consumes, and ensure that data is logically stored and is in third normal form (3NF). The database community has developed a series of guidelines or benchmarks for ensuring that databases are indeed normalized. These goals are referred to as normal forms and are numbered from one (the lowest form of normalization, referred to as first normal form or 1NF) through five (fifth normal form or 5NF). The first two normal forms are intermediate steps to achieve the goal of having all tables in 3NF. Note that in order to be correct, decomposition must be lossless i.e. the new tables can be recombined by a natural join to recreate the original table without creating any false or redundant data and without any loss of any data/information.

Consider a student database system to be developed for a multi-campus university, such that it specializes in one degree program at a campus i.e. BS, MS or PhD.



SID	Degree	Campus	Course	Marks
1	BS	Islamabad	CS-101	30
1	BS	Islamabad	CS-102	20
1	BS	Islamabad	CS-103	40
1	BS	Islamabad	CS-104	20
1	BS	Islamabad	CS-105	10
1	BS	Islamabad	CS-106	10
2	MS	Lahore	CS-101	30
2	MS	Lahore	CS-102	40
3	MS	Lahore	CS-102	20
4	BS	Islamabad	CS-102	20
4	BS	Islamabad	CS-104	30
4	BS	Islamabad	CS-105	40

SID: Student ID

Degree: Registered as BS or MS student

Campus: City where campus is located

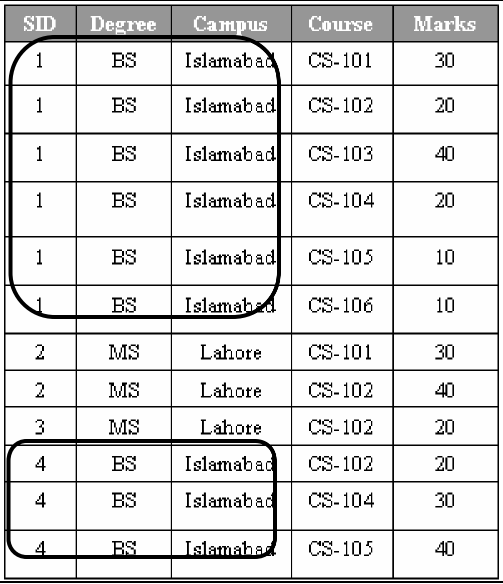
Course: Course taken

Marks: Score out of max of 50

Figure-6.1: Part of example student database system

We consider the case of development of a student DBMS for a multi-campus university i.e. with campuses in many cities. There is a head office of the University (say) in Islamabad which would like to get an overall picture of all the students at all the campuses of the university. A typical detail table is shown with a brief description of the fields. These are in no way the only fields, but sufficient to briefly review the basic forms of normalization.

In order to uniquely associate marks with course and the student, a composite primary key composed of SID and campus is used.



SID	Degree	Campus	Course	Marks
1	BS	Islamabad	CS-101	30
1	BS	Islamabad	CS-102	20
1	BS	Islamabad	CS-103	40
1	BS	Islamabad	CS-104	20
1	BS	Islamabad	CS-105	10
1	BS	Islamabad	CS-106	10
2	MS	Lahore	CS-101	30
2	MS	Lahore	CS-102	40
3	MS	Lahore	CS-102	20
4	BS	Islamabad	CS-102	20
4	BS	Islamabad	CS-104	30
4	BS	Islamabad	CS-105	40

Table FIRST

Figure-6.2: Part of example student database system in 1NF

Although the table is shown in 1NF i.e. it contains atomic values, there are no repeating values, there is no aggregation, yet it contains redundant data. For example, information about the student's degree, and campus location is repeated for every course taken. Redundancy causes what are called *update anomalies*. Update anomalies are problems that arise when records are inserted, deleted, or updated in the database. For example, the following anomalies can occur:

Normalization: 1NF
Update Anomalies

INSERT. Certain student with SID 5 got admission in a different campus (say) Karachi cannot be added until the student registers for a course.

DELETE. If student graduates and his/her corresponding record is deleted, then all information about that student is lost.

UPDATE. If student migrates from Islamabad campus to Lahore campus (say) SID = 1, then six rows would have to be updated with this new information.

INSERT. The fact that a certain student with SID 5 got admission in a different campus (say) Karachi cannot be added until the student registers for a course.

DELETE. If student graduates and his/her corresponding record is deleted, then all information about that student is lost.

UPDATE. If student migrates from Islamabad campus to Lahore campus (say) SID = 1, then six rows would have to be updated with this new information.

Normalization: 2NF

Every non-key column is fully dependent on the PK.

FIRST is in 1NF but not in 2NF because degree and campus are functionally dependent upon only on the column SID of the composite key (SID, course). This can be illustrated by listing the functional dependencies in the table:

$$\left.\begin{array}{l}\text{SID —> campus, degree}\\ \\ \text{campus —> degree}\\ \\ \text{(SID, Course) —> Marks}\end{array}\right\}\ \text{SID and Campus are NOT unique}$$

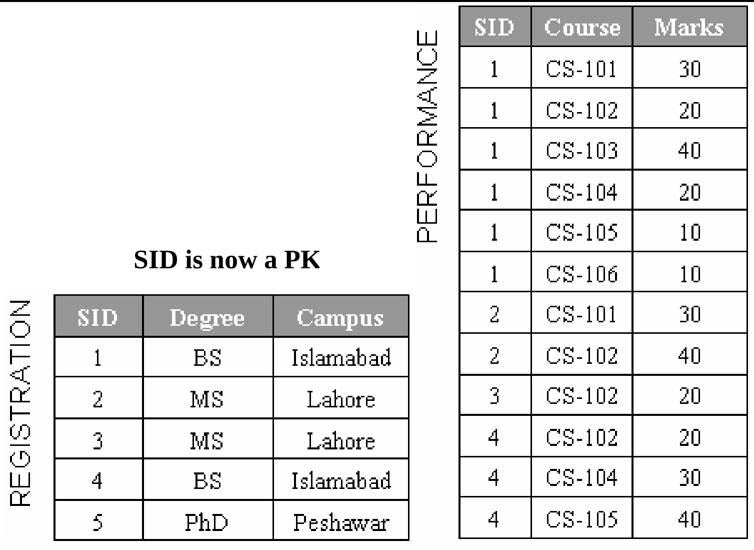
To transform the table FIRST into 2NF we move the columns SID, Degree and Campus to a new table called REGISTRATION. The column SID becomes the primary key of this new table.

The definition of second normal form states that only tables with composite primary keys can be in 1NF but not in 2NF.

A relational table is in second normal form 2NF if it is in 1NF and every non-key column is fully dependent upon the primary key. That is, every non-key column must be dependent upon the entire primary key. The table in the last slide is in 1NF but not in 2NF because degree, course and even marks are functionally dependent upon the SID and the campus.

FIRST is in 1NF but not in 2NF because degree and campus are functionally dependent upon only on the column SID of the composite key (SID, course). This can be illustrated by listing the functional dependencies in the table:

SID —> campus, status
campus —> degree
(SID, Course) —> Marks


PERFORMANCE
REGISTRATION

SID is now a PK

REGISTRATION

SID	Degree	Campus
1	BS	Islamabad
2	MS	Lahore
3	MS	Lahore
4	BS	Islamabad
5	PhD	Peshawar

PERFORMANCE

SID	Course	Marks
1	CS-101	30
1	CS-102	20
1	CS-103	40
1	CS-104	20
1	CS-105	10
1	CS-106	10
2	CS-101	30
2	CS-102	40
3	CS-102	20
4	CS-102	20
4	CS-104	30
4	CS-105	40

PERFORMANCE in 2NF as (SID, Course) uniquely identify Marks

Figure-6.3: Part of example student database system in 2NF

To transform the table FIRST into 2NF we move the columns SID, Degree and city to a new table called REGISTRATION. The column SID becomes the primary key of this new table.

Normalization: 2NF

Presence of modification anomalies for tables in 2NF. For the table REGISTRATION, they are:

- **INSERT:** Until a student gets registered in a degree program, that program cannot be offered!
- **DELETE:** Deleting any row from REGISTRATION destroys all other facts in the table.

Why there are anomalies?

The table is in 2NF but NOT in 3NF

Tables in 2NF but not in 3NF still contain modification anomalies. In the example of REGISTRATION, they are:

- INSERT. The fact that a particular campus has a certain degree (Peshawar campus decides to run a PhD program) cannot be inserted until there is a student registered for PhD in the campus.
- DELETE. Deleting any row from REGISTRATION table destroys the degree information about the campus as well as the association between student and campus.

Normalization: 3NF

All columns must be dependent only on the primary key.

Table PERFORMANCE is already in 3NF. The non-key column, marks, is fully dependent upon the primary key (SID, degree).

REGISTRATION is in 2NF but not in 3NF because it contains a transitive dependency.

A transitive dependency occurs when a non-key column that is a determinant of the primary key is the determinate of other columns.

The concept of a transitive dependency can be illustrated by showing the functional dependencies:

REGISTRATION.SID	—>	REGISTRATION.Degree
REGISTRATION.SID	—>	REGISTRATION.Campus
REGISTRATION.Campus	—>	REGISTRATION.Degree

Note that REGISTRATION.Degree is determined both by the primary key SID and the non-key column campus.

For a relational table to be in third normal form (3NF) all columns must be dependent only upon the primary key. More formally, a relational table is in 3NF if it is already in 2NF and every non-key column is non transitively dependent upon its primary key. In other words, all non-key attributes are functionally dependent only upon the primary key. Or put another way, all attributes must be directly dependent on the primary key without implied dependencies through other attributes of the relation.

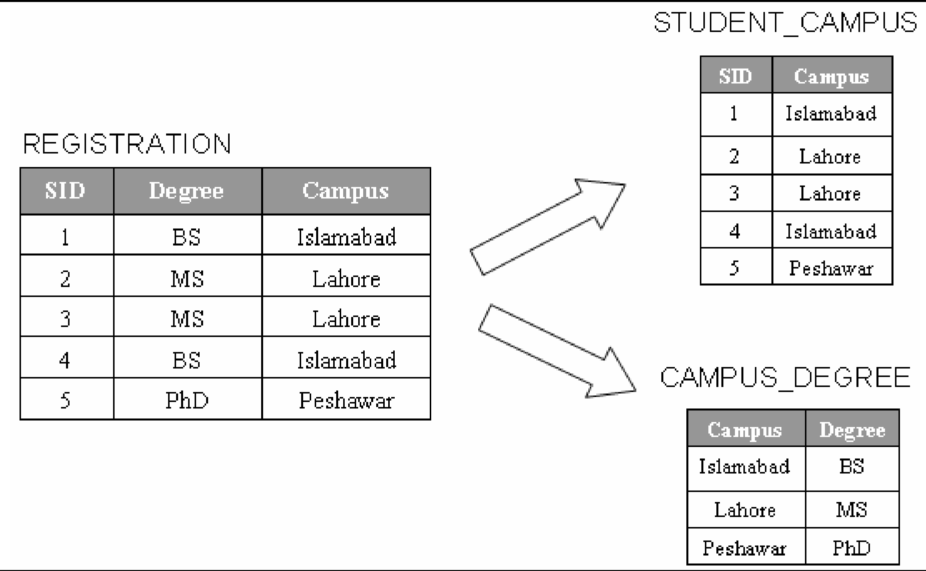
Table PERFORMANCE is already in 3NF. The non-key column marks is fully dependent and identified using the primary key SID and COURSE. However, table REGISTRATION is still only in 2NF as there is a *transitive dependency*. A transitive dependency arises when a non-key column that is a determinant of the primary key is the determinate of other columns.

To transform REGISTRATION into 3NF, a new table is created called CAMPUS_DEGREE and the columns campus and degree are moved into it. Degree is deleted from the original table, campus is left behind to serve as a foreign key to table CAMPUS_DEGREE, and the original table is renamed to STUDENT_CAMPUS to reflect its semantic meaning.

Normalization: 3NF

To transform REGISTRATION into 3NF, we create a new table called CAMPUS_DEGREE and move the columns campus and degree into it.

Degree is deleted from the original table, campus is left behind to serve as a foreign key to CAMPUS_DEGREE, and the original table is renamed to STUDENT_CAMPUS to reflect its semantic meaning.



REGISTRATION

SID	Degree	Campus
1	BS	Islamabad
2	MS	Lahore
3	MS	Lahore
4	BS	Islamabad
5	PhD	Peshawar

STUDENT_CAMPUS

SID	Campus
1	Islamabad
2	Lahore
3	Lahore
4	Islamabad
5	Peshawar

CAMPUS_DEGREE

Campus	Degree
Islamabad	BS
Lahore	MS
Peshawar	PhD

Figure-6.4: Part of example student database system in 3NF

Figure-6.4 shows that the table REGISTRATION is transformed into two tables i.e. Student_Campus and Campus_Degree. If we look at it in the context of memory, we observe that the storage space requirement has increased, for this particular example by about 7%.

Normalization: 3NF

Removal of anomalies and improvement in queries as follows:

- **INSERT:** Able to first offer a degree program, and then students registering in it.
- **UPDATE:** Migrating students between campuses by changing a single row.
- **DELETE:** Deleting information about a course, without deleting facts about all columns in the record.

Observe that by virtue of bringing a relational table into 3NF, storage of redundant data is further eliminated, that not only results in saving of space, but also reduces manipulation anomalies. For example, as a consequence following improvements have taken place:

INSERT: If Peshawar campus decides to offer a PhD program, this can be reflected even though there is no student currently registered in that campus. Similarly, facts about new students can be added even though they may not have registered for a course.

UPDATE: Changing the campus of a student (by migration) or a degree program of a campus requires modification of only a single row.

DELETE: Information about courses taken can be deleted without destroying information about a student or a campus.

Conclusions

- Normalization guidelines are cumulative.
- Generally a good idea to only ensure 2NF.
- 3NF is at the cost of s implicity and performance.
- There is a 4NF with no multi-valued dependencies.
- There is also a 5NF.

Normalization guidelines are cumulative. For a database to be in 2NF, it must first fulfill all the criteria of a 1NF database.

It is generally a good idea to ensure that the relational table is at least in 2NF. The goal is to avoid data redundancy so as to prevent slow corruption of data due to DML anomolies and make the best possible use of storage.

Although 3NF removes even more data redundancy, but this is at the cost of simplicity and performance. Hence it is upto the designer to find a balance between normalization and speed/simplicity.

There is a fourth normal form (4NF) with one additional requirement i.e. meet all the requirements of 3NF and there must be no multi-valued dependencies.

There is also a 5NF, but that is more of academic interest.

Lecture Handout

Data Warehousing

Lecture No. 07

De-Normalization

Striking a balance between "good" & "evil"

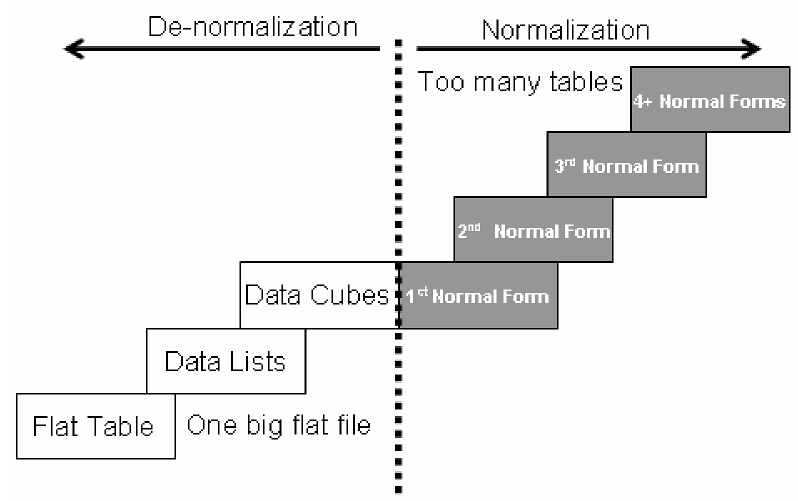


Figure-7.1: Striking a balance between normalization and de-normalization

There should be a balance between normalized and de-normalized forms. In a fully normalized form, too many joins are required and in a totally de-normalized form, we have a big, wide single table. Database should be aligned someplace in between so as to strike a balance, especially in the context of the queries and the application domain.

A "pure" normalized design is a good starting point for a data model and is a great thing to do and achieve in academia. However, as briefly mentioned in the previous lecture, in the reality of the "real world", the enhancement in performance delivered by some selective de-normalization technique can be a very valuable tool. The key to success is to undertake de-normalization as a design technique very cautiously and consciously. Do not let proliferation of the technique take over your data warehouse or you will end up with a single big flat file!

What is De-Normalization?

- It is not chaos, more like a "controlled crash" with the aim of performance enhancement without loss of information.
- Normalization is a rule of thumb in DBMS, but in DSS ease of use is achieved by way of denormalization.

- De-normalization comes in many flavors, such as combining tables, splitting tables, adding data etc., but all done very carefully.

'Denormalization' does not mean that anything and everything goes. Denormalization does not mean chaos or disorder or indiscipline. The development of properly denormalized data structures follows software engineering principles, which insure that information will not be lost. De-normalization is the process of selectively transforming normalized relations into un-normalized physical record specifications, with the aim of reducing query processing time. Another fundamental purpose of denormalization is to reduce the number of physical tables, that must be accessed to retrieve the desired data by reducing the number of joins required to answer a query. Some people tend to confuse dimensional modeling with de-normalization. This will become clear when we will cover dimensional modeling, where indeed tables are collapsed together.

Why De-Normalization in DSS?

- Bringing "close" dispersed but related data items.
- Query performance in DSS significantly dependent on physical data model.
- Very early studies showed performance difference in orders of magnitude for different number de-normalized tables and rows per table.
- The level of de-normalization should be carefully considered.

The efficient processing of data depends how close together the related data items are. Often all the attributes that appear within a relation are not used together, and data from different relations is needed together to answer a query or produce a report. Although normalized relations solve data maintenance anomalies (discussed in last lecture), however, normalized relations if implemented one for one as physical records, may not yield efficient data processing times. DSS query performance is a function of the performance of every component in the data delivery architecture, but is strongly associated with the physical data model. Intelligent data modeling through the use of techniques such as de-normalization, aggregation, and partitioning, can provide orders of magnitude performance gains compared to the use of normalized data structures.

The processing performance between totally normalized and partially normalized DSSs can be dramatic. Inmon (grand father of data warehousing) reported way back in 1988 through a study, by quantifying the performance of fully and partially normalized DSSs. In his study, a fully normalized DSS contained eight tables with about 50,000 rows each, another partially normalized DSS had four tables with roughly 25,000 rows each, and yet another partially normalized DSS had two tables. The results of the study showed that the less than fully normalized DSSs could muster a performance as much as an order of magnitude better than the fully normalized DSS. Although such results depend greatly on the DSS and the type of processing, yet these results suggest that one should carefully consider whether the physical records should exactly match the normalized relations for a DSS or not?

How De-Normalization improves performance?

De-normalization specifically improves performance by either:

- Reducing the number of tables and hence the reliance on joins, which consequently speeds up performance.
- Reducing the number of joins required during query execution, or
- Reducing the number of rows to be retrieved from the Primary Data Table.

The higher the level of normalization, the greater will be the number of tables in the DSS as the depth of snowflake schema would increase. The greater the number of tables in the DSS, the more joins are necessary for data manipulation. Joins slow performance, especially for very large tables for large data extractions, which is a norm in DSS not an exception. De-normalization reduces the number of tables and hence the reliance on joins, which consequently speeds up performance.

De-normalization can help minimize joins and foreign keys and help resolve aggregates. By storing values that would otherwise need to be retrieved (repeatedly), one may be able to reduce the number of indexes and even tables required to process queries.

4 Guidelines for De-normalization

1. Carefully do a cost-benefit analysis (frequency of use, additional storage, join time).

2. Do a data requirement and storage analysis.

3. Weigh against the maintenance issue of the redundant data (triggers used).

4. When in doubt, don't denormalize.

Guidelines for Denormalization:-
Following are some of the basic guidelines to help determine whether it's time to denormalize the DSS design or not:

1. Balance the frequency of use of the data items in question, the cost of additional storage to duplicate the data, and the acquisition time of the join.

2. Understand how much data is involved in the typical query; the amount of data affects the amount of redundancy and additional storage requirements.

3. Remember that redundant data is a performance benefit at query time, but is a performance liability at update time because each copy of the data needs to be kept up to date. Typically triggers are written to maintain the integrity of the duplicated data.

4. De-normalization usually speeds up data retrieval, but it can slow the data modification processes. It may be noted that both on-line and batch system performance is adversely affected by a high degree of de-normalization. Hence the golden rule is: When in doubt, don't denormalize.

Areas for Applying De-Normalization Techniques

- Dealing with the abundance of star schemas.
- Fast access of time series data for analysis.
    - Fast aggregate (sum, average etc.) results and complicated calculations.
    - Multidimensional analysis (e.g. geography) in a complex hierarchy.
    - Dealing with few updates but many join queries.

De-normalization will ultimately affect the database size and query performance.

De-normalization is especially useful while dealing with the abundance of star schemas that are found in many data warehouse installations. For such cases, de-normalization provides better performance and a more natural data structure for supporting decision making. The goal of most analytical processes in a typical data warehouse environment is to access aggregates such as averages, sums, complicated formula calculations, top 10 customers etc. Typical OLTP systems contain only the raw transaction data, while decision makers expect to find aggregated and time-series data in their data warehouse to get the big picture through immediate query and display. Important and common parts of a data warehouse that are a good candidate for de-normalization include (but are not limited to):

- Aggregation and complicated calculations.
- Multidimensional analysis in a complex hierarchy.
- Few updates, but many join queries.

Geography is a good example of a multidimensional hierarchy (e.g. Province, Division, District, city and zone). Basic design decisions for example, the selection of dimensions, the number of dimensions to be used and what facts to aggregate will ultimately affect the database size and query performance.

Five principal De-normalization techniques

1. Collapsing Tables.
    - Two entities with a One-to-One relationship.
    - Two entities with a Many-to-Many relationship.

2. Pre-Joining.

3. Splitting Tables (Horizontal/Vertical Splitting).

4. Adding Redundant Columns (Reference Data).

5. Derived Attributes (Summary, Total, Balance etc).

Now we will discuss de-normalization techniques that have been commonly adopted by experienced database designers. These techniques can be classified into four prevalent strategies for denormalization which are:

1. Collapsing Tables.
- Two entities with a One-to-One relationship.

- Two entities with a Many-to-Many relationship.

2. Pre-joining.

3. Splitting Tables (Horizontal/Vertical Splitting).

4. Adding Redundant Columns (Reference Data).

5. Derived Attributes (Summary, Total, Balance etc).

Collapsing Tables

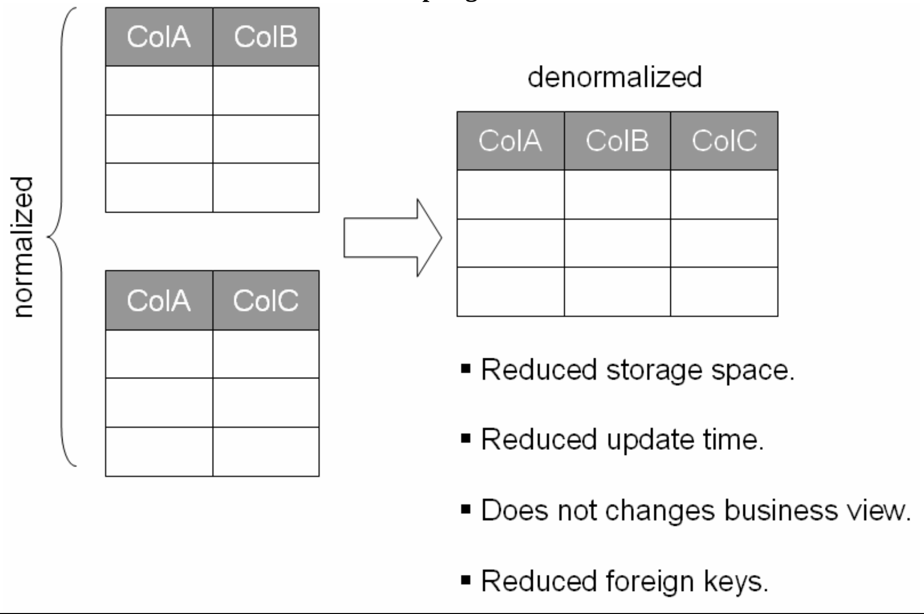


Figure-7.1: Collapsing Tables

1. Collapsing Tables

One of the most common and safe denormalization techniques is combining of One-to-One relationships. This situation occurs when for each row of entity A, there is only one related row in entity B. While the key attributes for the entities may or may not be the same, their equal participation in a relationship indicates that they can be treated as a single unit. For example, if users frequently need to see COLA, COLB, and COLC together and the data from the two tables are in a One-to-One relationship, the solution is to collapse the two tables into one. For example, SID and gender in one table, and SID and degree in the other table.

In general, collapsing tables in One-to-One relationship has fewer drawbacks than others. There are several advantages of this technique, some of the obvious ones being reduced storage space, reduced amount of time for data update, some of the other not so apparent advantages are reduced number of foreign keys on tables, reduced number of indexes (since most indexes are created based on primary/foreign keys). Furthermore, combining the columns does not change the business view, but does decrease access time by having fewer physical objects and reducing overhead.

Lecture Handout

Data Warehousing

Lecture No. 08

De-Normalization Techniques

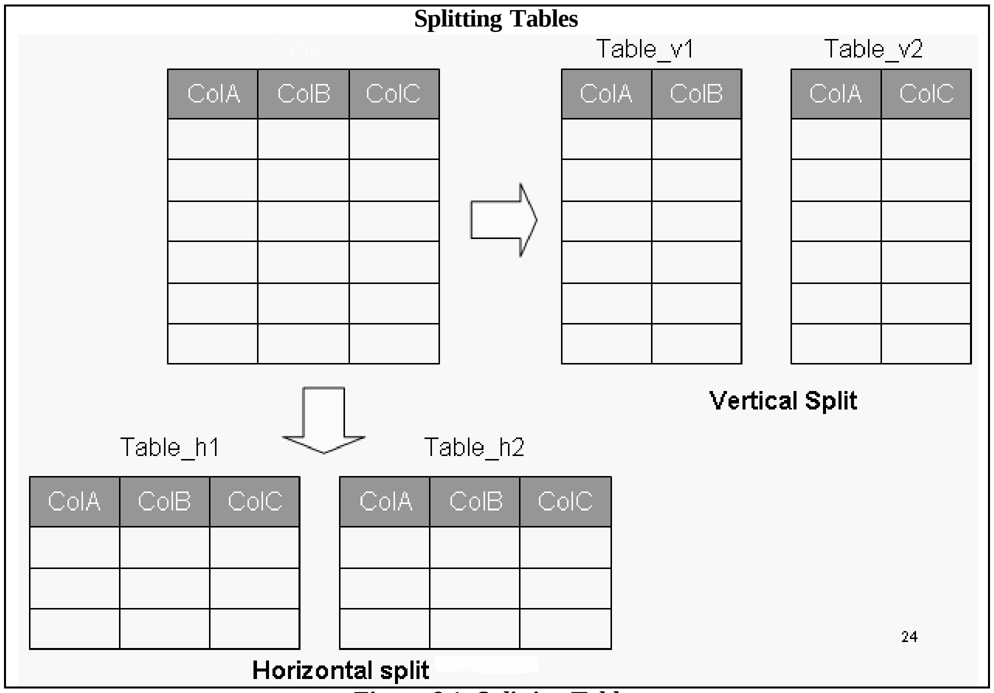


Figure-8.1: Splitting Tables

Splitting Tables

The denormalization techniques discussed earlier all dealt with combining tables to avoid doing run-time joins by decreasing the number of tables. In contrast, denormalization can be used to create more tables by splitting a relation into multiple tables. Both horizontal and vertical splitting and their combination are possible. This form of denormalization -record splitting- is especially common for a distributed DSS environment.

Splitting Tables: Horizontal splitting

Breaks a table into multiple tables based upon common column values. Example: Campus specific queries.

GOAL

- Spreading rows for exploiting parallelism.
- Grouping data to avoid unnecessary query load in WHERE clause.

Splitting Tables

Horizontal splitting breaks a relation into multiple record set specifications by placing different rows into different tables based upon common column values. For the multi-campus example being considered; students from Islamabad campus in the Islamabad table, Peshawar students in corresponding table etc. Each file or table created from the splitting has the same record lay out or header.

Goals of Horizontal splitting:

There are typically two specific goals for horizontal partitioning: (1) spread rows in a large table across many HW components (disks, controllers, CPUs, etc.) in the environment to facilitate parallel processing, and (2) segregate data into separate partitions so that queries do not need to examine all data in a table when WHERE clause filters specify only a subset of the partitions. Of course, what we would like in an ideal deployment is to get both of these benefits from table partitioning.

Advantages of Splitting tables:

Horizontal splitting makes sense when different categories of rows of a table are processed separately: e.g. for the student table if a high percentage of queries are focused towards a certain campus at a time then the table is split accordingly. Horizontal splitting can also be more secure since file level of security can be used to prohibit users from seeing certain rows of data. Also each split table can be organized differently, appropriate for how it is individually used. In terms of page access, horizontally portioned files are likely to be retrieved faster as compared to un-split files, because the latter will involve more blocks to be accessed.

Splitting Tables: Horizontal splitting

ADVANTAGE

- Enhance security of data.
- Organizing tables differently for different queries.
- Reduced I/O overhead.
- Graceful degradation of database in case of table damage.
- Fewer rows result in flatter B-trees and fast data retrieval.

Other than performance, there are some other very useful results of horizontal splitting the tables. As we discussed in the OLAP lecture, security is one of the key features required from an OLAP system. Actually DSS is a multi-user environment, and robust security needs to ensure. By splitting the tables and restricting the users to a particular split actually improves the security of the system. Consider time-based queries, if the queries have to cover last years worth of data, then splitting the tables on the basis of year will defiantly improve the performance as the amount of data to be accessed is reduced. Similarly, if for a multi-campus university, most of the queries are campus specific, then splitting the tables based on the campus would result in improved performance. In both of the cases of splitting discussed i.e. time and space, as the number of records to be retrieved is reduced, resulting in more records per block that translates into fewer page faults and high performance. If the table is not partitioned, and for some reason the table is damaged, then in the worst case all data might be lost. However, when the table gets partitioned, and even if a partition is damaged, ALL of the data is not lost. Assuming a worst case scenario that tables crash i.e. all of them, the system will not go down suddenly, but would go down gradually i.e. gracefully.

Splitting Tables: Vertical Splitting

- Splitting and distributing into separate files with repeating primary key.
- Infrequently accessed columns become extra "baggage" thus degrading performance.
- Very useful for rarely accessed large text columns with large headers.
- Header size is reduced, allowing more rows per block, thus reducing I/O.
- For an end user, the split appears as a single table through a view.

Splitting Tables: Vertical Splitting

Vertical splitting involves splitting a table by columns so that a group of columns is placed into the new table and the remaining columns are placed in another new table. Thus columns are distributed into separate files, such that the primary key is repeated in each of the files. An example of vertical splitting would be breaking apart the student registration table by creating a personal_info table by placing SID along with corresponding data into one record specification, the SID along with demographic-related student data into another record specification, and so on.

Vertical splitting can be used when some columns are rarely accessed rather than other columns or when the table has wide rows or header or both. Thus the infrequently accessed columns become extra "baggage" degrading performance. The net result of splitting a table is that it may reduce the number of pages/blocks that need to be read because of the shorter header length allowing more rows to be packed in a block, thus reducing I/O. A vertically split table should contain one row per primary key in the split tables as this facilitates data retrieval across tables and also helps in dissolving the split i.e. making it reversible. In reality, the users are unaware of the split, as view of a joined table is presented to the users.

2. Pre-Joining

Pre-Joining

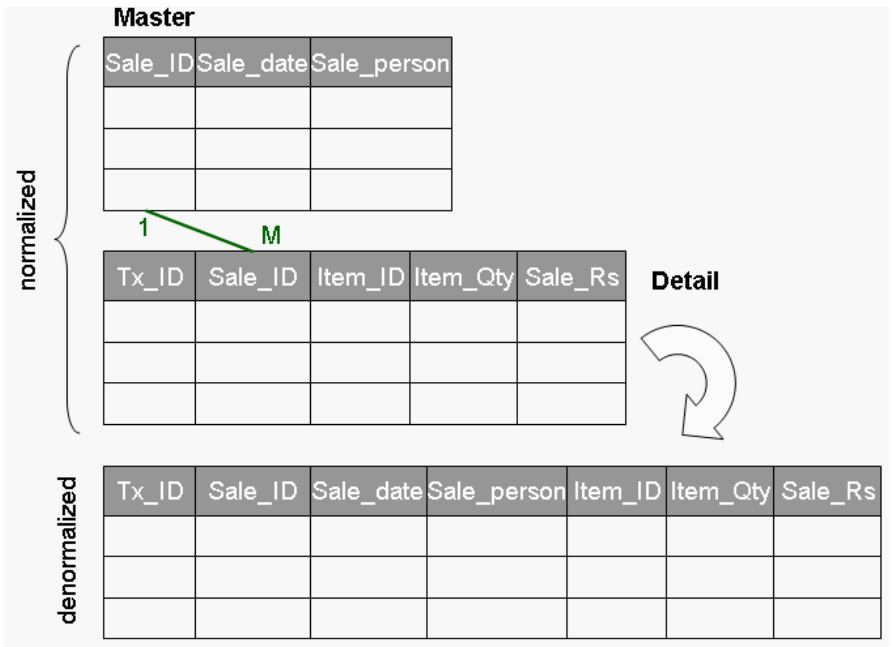


Figure-8.2: Pre-joining Tables

The objective behind pre-joining is to identify frequent joins and append the corresponding tables together in the physical data model. This technique is generally used when there is a one-to-many relationship between two (or more) tables, such as the master-detail case when there are header and detail tables in the logical data model. Typically, referential integrity is assumed from the foreign key in one table (detail) to the primary key in the other table (header).

Additional space will be required, because information on the master table is stored once for each detail record i.e. multiple times instead of just once as would be the case in a normalized design.

Pre-Joining

- Typical of Market basket query
- Join ALWAYS required
- Tables could be millions of rows
- Squeeze Master into Detail
- Repetition of facts. How much?
- Detail 3-4 times of master

Figure-8.2 shows a typical case of market basket querying, with a master table and a detail table. The sale_ID column is the primary key for the master table and uniquely identifies a market basket. There will be one "detail" record for each item listed on the "receipt" for the market basket. The tx_ID column is the primary key for the detail table.

Observe that in a normalized design the store and sale date for the market basket is on one table (master) and the item (along with quantity, sales Rs, etc.) are on a separate table (detail). Almost all analysis will require product, sales date, and (sometimes) sale person (in the context of HR). Moreover, both tables can easily be millions of rows for a large retail outlet with significant historical data. This means that a join will be forced between two very large tables for almost every query asked of the data warehouse. This could easily choke the system and degrade performance.

Note that this same header/detail structure in the data applies across many industries as we have discussed in the very early lectures, such as healthcare, transportation, logistics, billing etc.

To avoid the run-time join, we use the pre-join technique and "squeeze" the sales master information into the detail table. The obvious drawback is repetition of facts from the master table into the detail table. This avoids the join operation at run-time, but stores the header information redundantly as part of the sales detail. This redundant storage is a violation of normalization, but will be acceptable if the cost of storage is less then the performance achieved by virtue of eliminating the join.

4. Adding Redundant Columns

This technique can be used when a column from one table is frequently accessed in a large scale join in conjunction with a column from another table. To avoid this join, the column is added (redundant) or moved into the detail table(s) to avoid the join. For example, if frequent joins are performed using Table_1 and Table_2 using columns ColA, ColB and ColC, then it is suitable to add ColC to Table_1.

Adding Redundant Columns

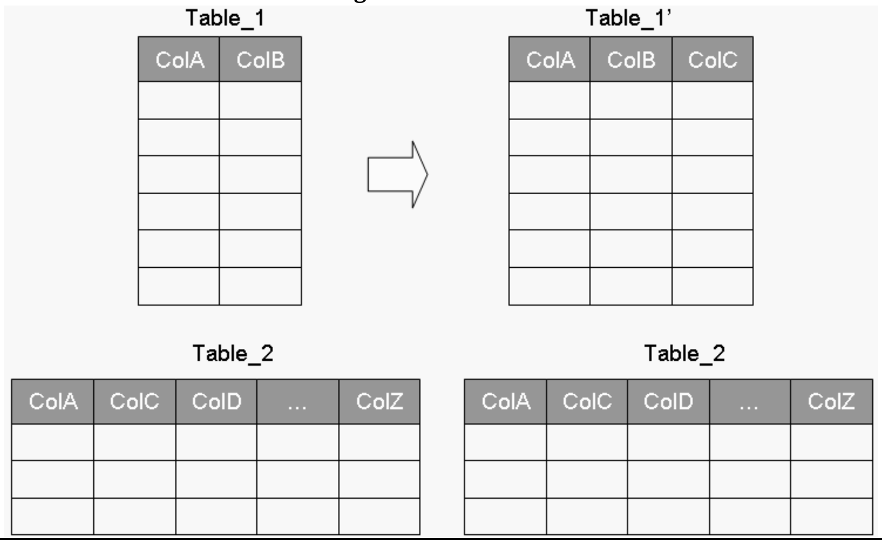


Figure-8.3: Adding redundant columns

Note that the columns can also be moved, instead of making them redundant. If closely observed, this technique is no different from a pre-joining. In pre-joining all columns are moved from the master table into the detail table, but in the current case, a sub-set of columns from the master table is made redundant or moved into the detail table. The performance, and storage trade-offs are also very similar to pre-joining.

Adding Redundant Columns

Columns can also be moved, instead of making them redundant. Very similar to pre-joining as discussed earlier.

EXAMPLE
Frequent referencing of code in one table and corresponding description in another table.

- A join required is required.
- To eliminate the join, a redundant attribute added in the target entity which is functionally <u>independent</u> of the primary key.

A typical scenario for column redundancy/movement is frequent referencing of code in one table and the corresponding description in another table. The description of a code is retrieved via a join. In such a case, redundancy will naturally pay off. This is implemented by duplicating the descriptive attribute in the entity, which would otherwise contain only the code. The result is a redundant attribute in the target entity which is functionally independent of the primary key. Note that this foreign key relationship was created in the first place to normalize the corresponding description reduce update anomalies.

Redundant Columns: Surprise

Note that:

- Actually increases in storage space, and increase in update overhead.
- Keeping the actual table intactand unchanged helps enforce RI constraint.
- Age old debate of RI ON or OFF.

Redundant Columns Surprise

Creating redundant columns does not necessarily reduce the storage space requirements, as neither the reference table is removed, nor the columns duplicated from the reference table.

The reason being to ensure data input RI constraint, although this reasoning falls right in the middle of the age old debate that Referential Integrity (RI) constraint should be turned ON or OFF in a DWH environment. However, it is obvious that column redundancy does eliminate the join and increase the performance.

Derived Attributes

- Objectives
  - Ease of use for decision support applications
  - Fast response to predefined user queries
  - Customized data for particular target audiences
  - Ad-hoc query support
- Feasible when…
  - Calculated once, used most
  - Remains fairly "constant"
  - Looking for absoluteness of correctness.
  - Pitfall of additional space and query degradation.

5. Derived Attributes

It is usually feasible to add derived attribute(s) in the data warehouse data model, if the derived data is frequently accessed and calculated once and is fairly stable. The justification of adding derived data is simple; it reduces the amount of query processing time at run-time while accessing the data in the warehouse. Furthermore, once the data is properly calculated, there is little or no apprehension about the authenticity of the calculation. Put in other words, once the derived data is properly calculated it kind of becomes absolute i.e. there is hardly any chance that someone might use a wrong formula to calculate it incorrectly. This actually enhances the credibility of the data in the data warehouse.

Derived Attributes

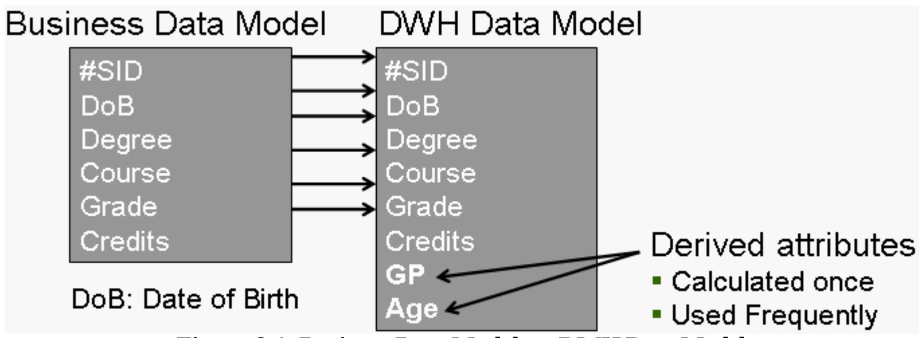


Figure-8.4: Business Data Model vs. DWH Data Model

GP (Grade Point) column in the data warehouse data model is included as a derived value. The formula for calculating this field is Grade*Credits.

Age is also a derived attribute, calculated as Current_Date – DoB (calculated periodically).

In most cases, it will only make sense to use derived data if the ratio of detail rows to derived rows is at least 10:1. In such cases, the 10% storage cost for keeping the derived data is less than the temporary and sort space storage costs for many concurrent queries aggregating at runtime.

Lecture Handout

Data Warehousing

Lecture No. 09

Issues of De-Normalization

- Storage
- Performance
- Maintenance
- Ease-of-use

The effects of denormalization on database performance are unpredictable: as many applications/users can be affected negatively by denormalization when some applications are optimized. If a decision is made to denormalize, make sure that the logical model has been fully normalized to 3NF. Also document the pure logical model and keep your documentation of the physical model current as well. Consider the following list of effects of denormalization before you decide to undertake design changes.

The trade-offs of denormalization are as follows:

- Storage
- Performance
- Ease-of-use
- Maintenance

Each of these tradeoffs must be considered when deploying denormalization into a physical design. Typically, architects are pretty good at assessing performance and storage implications of a denormalization decision. Factors that are notoriously under estimated are the maintenance implications and the impact on usability/flexibility for the physical data model.

Storage Issues: Pre-joining

- Assume 1:2 record count ratio between claim master and detail for health-care application.
- Assume 10 million members (20 million records in claim detail).
- Assume 10 byte member_ID.
- Assume 40 byte header for master and 60 byte header for detail tables.

By understanding the characteristics of the data, the storage requirements can actually be quantified before pre-joining. We need to know the size of the data from the master table that will be replicated for pre-joining into the detail table as well as the number of detail records (on average) in the header that will be denormalized as a result of pre-joining.

In this example, it is assumed that that each master table record has two detail record entries associated with it (on average). Note that this ratio will vary depending on the nature of each industry and business within an industry. The health-care industry would be much closer to a 1:2 ratio, depending on if the data is biased towards individual or organizational claims. A 1:3 ratio could be reasonable for a video rental store, but a grocery store with tens of thousands of items, the ratio would typically be on the plus side of 1:30 detail records for each master table entry. It is important to know the characteristics in your specific environment to properly and correctly calculate the storage requirements of the pre-joining technique.

Storage Issues: Pre-joining

<u>With normalization</u>:
Total space used = 10 x 40 + 20 x 60 = 1.6 GB

<u>After denormalization:</u>
Total space used = (60 + 40 – 10) x 20 = 1.8 GB

Net result is 12.5% additional space required in raw data table size for the database.

The 12.5% investment in additional storage for pre-joining will dramatically increase performance for queries which would otherwise need to join the very large header and detail tables.

Performance Issues: Pre-joining

Consider the query "How many members were paid claims during last year?"

<u>With normalization:</u>
Simply count the number of records in the master table.

<u>After denormalization:</u>
The member_ID would be repeated, hence need a count distinct. This will cause sorting on a larger table and degraded performance.

How the corresponding query will perform with normalization and after denormalization? This a good question, with a surprising answer. Observe that with normalization there are unique values in the master table, and the number of records in the master table is the required answer. To get this answer, there is probably no need to touch that table, as the said information can be picked from the meta-data corresponding to that table. However, it is a different situation after pre-joining has been performed. Now there are multiple i.e. repeating member_IDs in the joined table. Thus accessing the meta-data is not going to help. The only viable option is to perform a count distinct, easier said than done. The reason being this will require a sort operation, and then dropping the repeating value. For large tables, it is going to kill the performance of the system.

Performance Issues: Pre-joining

Depending on the query, the performance actually deteriorates with denorma lization! This is due to the following three reasons:

- Forcing a sort due to count distinct.
- Using a table with 2.5 times header size.
- Using a table which is 2 times larger.
- Resulting in 5 times degradation in performance.

Bottom Line: Other than 0.2 GB additional space, also keep the 0.4 GB master table.

Counter intuitively, the query with pre-joining will perform worse than the normalized design, basically for three reasons.

1. There is no simple way to count the number of distinct customers in the physical data model because this information is now "lost" in the detail table. As a result, there is no choice, but to use a "count distinct" on the member_ID to group identical IDs and then determine the number of unique patients. This is going to be achieved by sorting all qualifying rows (by date) in the denormalized tables. Note that sorting is typically a very expensive operation, the best being O(n log n).

2. The table header of the denormalized detail table is now 90 bytes as opposed to 40 bytes of the master table i.e. an increase of 250%.

3. The number of rows that need to be scanned in the details table are two times as many as compared to the normalized design i.e. scanning the master table. This translates to five times more I/Os in the denormalized scenario versus the normalized scenario!

Bottom line is that the normalized design is likely to perform many times faster as compared to the denormalized design for queries that probe the master table alone, rather than those that perform a join between the master and the detail table. Best and expensive approach would be to also keep the normalized master table, and a smart query coordinator that directs the queries to for increasing performance.

Performance Issues: Adding redundant columns

Continuing with the previous Health-Care example, assuming a 60 byte detail table and 10 byte Sale_Person.

- Copying the Sale_Person to the detail table results in all scans taking 16% longer than previously.

- Justifiable only if significant portion of queries get benefit by accessing the denormalized detail table.

- Need to look at the cost-benefit trade-off for each denormalization decision.

The main problem with redundant columns is that if strict discipline is not enforced, it can very quickly result into chaos. The reason being, every DWH user has their own set of columns which they frequently use in their queries. Once they hear about the performance benefits (due to denormalization) they would want their "favorite" column(s) to be moved/copied into the main fact table in the data warehouse. If this is allowed to happen, sooner than later the fact table would become one large flat file with a header in kilo bytes, and result in degraded performance.

The reason being, each time the table width is increased, the number of rows per block decreases and the amount of I/O increases, and the table access becomes less efficient. Hence the column redundancy can not be looked into isolation, with a view to benefit a only a subset of the queries. For a number of queries, the performance will degrade by avoiding the join, thus a detailed and quantifiable cost-benefit analysis is required.

Other Issues: Adding redundant columns

Other issues include, increase in table size, maintenance and loss of information:

- The size of the (largest table i.e.) transaction table increases by the size of the Sale_Person key.
    - For the example being considered, the detail table size increases from 1.2 GB to 1.32 GB.
- If the Sale_Person key changes (e.g. new 12 digit NID), then updates to be reflected all the way to transaction table.
- In the absence of 1:M relationship, column movement will actually result in loss of data.

Maintenance is usually overlooked or underestimated while replicating columns. Because the cost of reflecting the change in the member_ID for this design, is considerably high when reflected across the relevant tables. For example, transactions in the detail table need to be updated with the new key, and for a retail warehouse, the detail table could be 30 times larger than the master table, which again is larger then the fact (member table). For an archival system that keeps backup of the historical transactions, maintenance becomes a nightmare, because keeping the member_id data consistent will be very risky.

Ease of use Issues: Horizontal Splitting

Horizontal splitting is a Divide&Conquer technique that exploits parallelism. The conquer part of the technique is about combining the results.

Lets see how it works for hash based splitting/partitioning.

- Assuming uniform hashing, hash splitting supports even data distribution across all partitions in a pre-defined manner.
- However, hash based splitting is not easily reversible to eliminate the split.

Hash partitioning is the most common partitioning strategy. Almost all parallel RDBMS products provide some form of built-in hash partitioning capability (mainframe DB2 is the most significant exception to this statement). Horizontal partitioning using a hashing algorithm will assign data rows to partitions according to a "repeatable" random algorithm. In other words, a particular row will always hash to the same partition (assuming that the hashing algorithm and number of partitions have not changed), but a large number of rows will be "randomly" distributed across the partitions as long as a well-selected partitioning key is selected and the hashing function is well-behaved.

Notice that the "random" assignment of data rows across partitions makes it nearly impossible to get any kind of meaningful partition elimination. Since data rows are hash distributed across all partitions (for load-balancing purposes), there is not practical way to perform partition elimination unless a very small number (e.g., singleton) or data rows is selected from the table via the partitioning key (which doesn't happen often in a traditional DW workload).

Ease of Use Issues: Horizontal Splitting

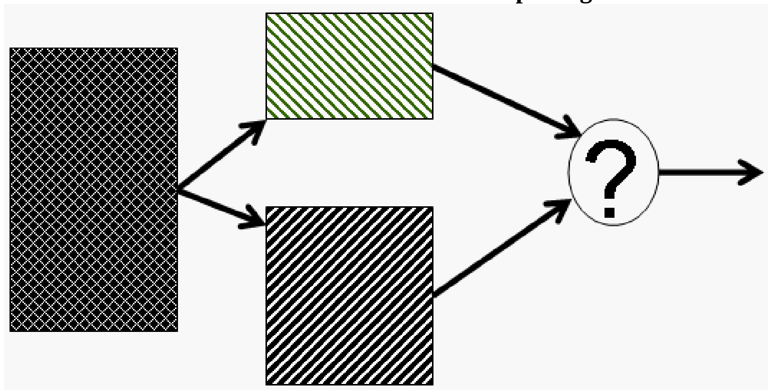


Figure-9.1: Irreversible partitioning

Note that we perform denormalization to get performance for a particular set of queries, and may like to bring the table back to its original form for another set of queries. If this can not be done, then extra effort or CPU cycles would be required to achieve this objective. As shown in Figure-9.1, it is possible to have a partitioning strategy, such that the partitioned tables can not be appended together to get the record sin the original order. This is further explained when we discuss the issues of horizontal partitioning.

Ease of Use Issues: Horizontal Splitting

- Round robin and random splitting:
  - Guarantee good data distribution.
  - Not pre-defined.
  - Almost impossible to reverse (or undo).

- Range and expression splitting:
  - Can facilitate partition elimination with a smart optimizer.
  - Generally lead to "hot spots" (uneven distribution of data).

Round-robin spreads data evenly across the partitions, but does not facilitate partition elimination (for the same reasons that hashing does not facilitate partition elimination). Round-robin is typically used only for temporary tables where partition elimination is not important and co-location of the table with other tables is not expected to yield performance benefits (hashing allows for co-location, but round-robin does not). Round-robin is the “cheapest” partitioning algorithm that guarantees an even distribution of workload across the table partitions.

The most common use of range partitioning is on date. This is especially true in data warehouse deployments where large amounts of historical data are often retained. Hot spots typically surface when using date range partitioning because the most recent data tends to be accessed most frequently.

Expression partitioning is usually deployed when expressions can be used to group data together in such a way that access can be targeted to a small set of partitions for a significant portion of the DW workload. For example, partitioning a table by expression so that data corresponding to different divisions or LOBs (lines of business) is grouped together will avoid scanning data in divisions or LOBs excluded from the WHERE clause predicates in a DSS query. Expression partitioning can lead to hot spots in the same way as described for range partitioning.

Performance Issues: Horizontal Splitting

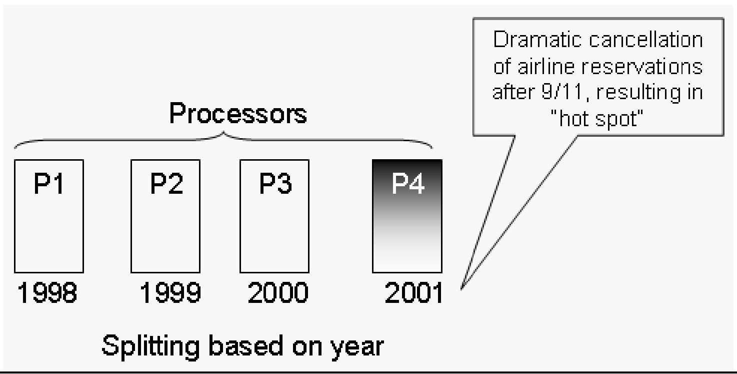

Dramatic cancellation of airline reservations after 9/11, resulting in "hot spot"
Processors
P1
P2
P3
P4
1998
1999
2000
2001
Splitting based on year

Figure-9.2: De-mertis of horizontal partitoiniong

Recall in last lecture when we discussed partitioning on the basis of date to enhance query performance, this has its down sides too. Consider the case of airline reservations table horizontally split on the basis of year. After 9/11 people obviously got scared of flying, and there was a surge of cancellations of air line bookings. Thus the most number of cancellations, actually, probably highest ever occurred during the last quarter of year 2001. Thus the corresponding partition would have the largest number of records. Thus in a parallel processing environment, where partitioning is consciously done to improve performance it not going to work, because as shown in Figure-9.2 most of the work is being done by processor P4 which would become a bottleneck. Meaning, unless the results of processor P4 are made available, the overall results can not be combined, while the remaining processors are idling i.e. doing nothing.

Performance issues: Vertical Splitting

Example: Consider a 100 byte header for the member table such that 20 bytes provide complete coverage for 90% of the queries.

Split the member table into two parts as follows:

1. Frequently accessed portion of table (20 bytes), and

2. Infrequently accessed portion of table (80+ bytes). Why 80+?

Note that primary key (member_id) must be present in both tables for eliminating the split.

Note that there will be a one-to-one relationship between rows in the two portions of the partitioned table.

Performance issues: Vertical Splitting

Scanning the claim table for most frequently used queries will be 500% faster with vertical splitting

Ironically, for the "infrequently" accessed queries the performance will be inferior as compared to the un-split table because of the join overhead.

Scanning the vertically partitioned claim table for frequently accessed data is five times faster than before splitting because the table is five times "thinner" without the infrequently used portions of the data.

However, this performance benefit will only be obtained if we do not have to join to the infrequently accessed portion of the table. In other words, all columns that we need must be in the frequently accessed portion of the table.

Performance issues: Vertical Splitting

Carefully identify and select the columns that get placed on which "side" of the frequently/infrequently used "divide" between the splits.

Moving a single five byte column to the frequently used table split (20 byte width) means that ALL table scans against the frequently used table will run 25% slower.

Don't forget the additional space required for the join key, this becomes significant for a billion row table.

Also, be careful when determining frequency of use. You may have 90% of the queries accessing columns in the "frequently used" partition of the table. However, the important measure is the percent of queries that access only the frequently used portion of the table with no columns required from the infrequently used data.

Lecture Handout

Data Warehousing

Lecture No. 10

Online Analytical Processing (OLAP)

OLAP is Analytical Processing instead of Transaction Processing. It is also NOT a physical database design or implementation technique, but a framework. Furthermore OLAP implementations are either highly de-normalized or completely de-normalized, while OLTP implementations are fully normalized. Therefore, a clear understanding of how OLAP (On-Line Analytical Processing) fits into the overall architecture of your data warehouse is essential for appropriate deployment. There are different methods of deployment of OLAP within a decision support environment - often obscured by vendor hype and product salesmanship. It is important that you as data warehouse architect drive your choice of OLAP tools - not driven by it!

DWH & OLAP

- Relationship between DWH & OLAP
- Data Warehouse & OLAP go together.
- Analysis supported by OLAP

A data warehouse is a "subject-oriented, integrated, time varying, non-volatile collection of data that is used primarily in organizational decision making. Typically, the data warehouse is maintained separately from the organization's operational databases and on different (and more powerful) servers. There are many reasons for doing this. The data warehouse supports on-line analytical processing (OLAP), the functional and performance requirements of which are quite different from those of the on-line transaction processing (OLTP) applications, traditionally supported by the operational data bases.

A data warehouse implementation without an OLAP tool is nearly unthinkable. Data warehousing and on-line analytical processing (OLAP) are essential elements of decision support, and go hand-in-hand. Many commercial OLAP products and services are available, and all of the principal database management system vendors have products in these areas. Decision support systems are very different from MIS systems, consequently this places some rather different requirements on the database technology compared to traditional on-line transaction processing applications.

Data Warehouse provides the best support for analysis while OLAP carries out the analysis task. Although there is more to OLAP technology than data warehousing, the classic statement of OLAP is "decision making is an iterative process; which must involve the users". This statement implicitly points to the not so obvious fact that while OLTP systems keep the users' records and assist the users in performing the routine operations tasks of their daily work. OLAP systems on the other hand generate information users require to do the non-routine parts of their jobs. OLAP systems deal with unpredicted circumstances (for which if not impossible, it is very difficult to plan ahead). If the data warehouse doesn't provide users with helpful information quickly in an intuitive format, the users would be turned down by it, and will revert back to their old and faithful, yet inefficient but tried out procedures. Data warehouse developers sometimes refer to the importance of intuitiveness to the users as the "field of dreams scenario" i.e. naively assuming that just because you build it doesn't mean they will come and use it.

Supporting the human thought process

We will first learn the nature of analysis before going on to the actual structure and functionality behind OLAP systems; this is discussed in Figure-10.1. Consider a hypothetical (but a realistic scenario) when a decision maker (analyst, CEO) of a large enterprise i.e. a multinational company is confronted with the yearly sales figure that there is an enterprise wide fall in profit. Note that we are talking about a large multinational company, which is not only in sales, but also in production. To get a hold of the problem, decision maker has to start from somewhere, but that somewhere has to be meaningful too. This "somewhere" could be starting the exploration across the enterprise geographically or exploring across the enterprise functionally or doing it product-wise or doing it with reference to time.

One of the valid options of starting is with reference to time, and the query corresponding to the thought process of the decision maker turns out to be "What was the quarterly sales during last year?" The result of this query would be sales figures for the four quarters of last year. There can be a number of possible outcomes, for example, the sale could be down in all the quarters, or it could be down in three out of four quarters with four possible options. Another possibility could be sales down in any two quarters with six possible scenarios. For the sake of simplicity, over here we assume that the sales were down in only one quarter. If the results were displayed as histograms, the decision maker would immediately pick the quarter in which sales were down, which in this case comes out to be the last quarter.

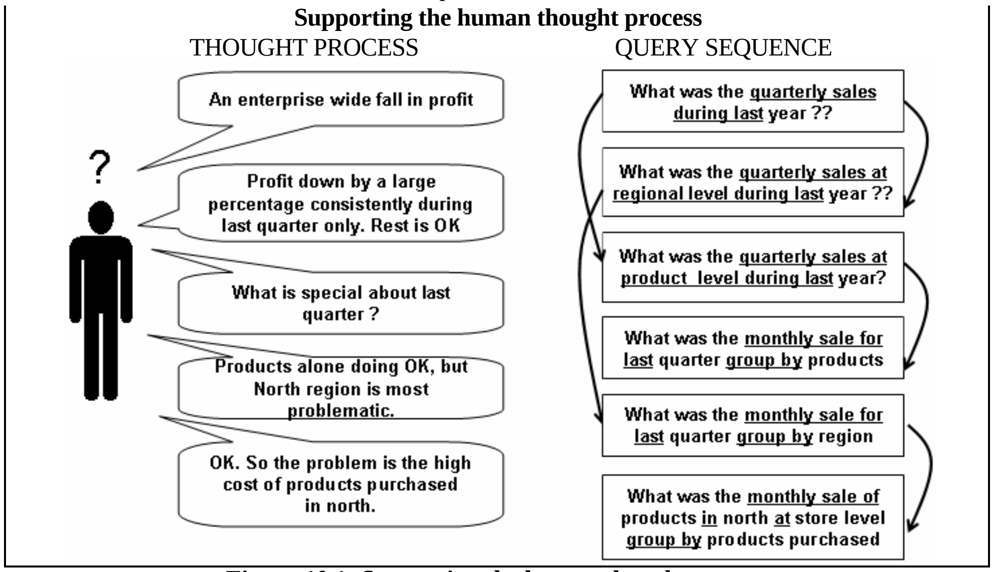

Supporting the human thought process
THOUGHT PROCESS
QUERY SEQUENCE
An enterprise wide fall in profit
?
Profit down by a large percentage consistently during last quarter only. Rest is OK
What is special about last quarter ?
Products alone doing OK, but North region is most problematic.
OK. So the problem is the high cost of products purchased in north.
What was the quarterly sales during last year ??
What was the quarterly sales at regional level during last year ??
What was the quarterly sales at product level during last year?
What was the monthly sale for last quarter group by products
What was the monthly sale for last quarter group by region
What was the monthly sale of products in north at store level group by products purchased

Figure-10.1: Supporting the human thought process

Now that the "lossy" quarter has been identified, the decision maker would naturally like to probe further, to go to the root of the problem, and would like to know what happened during the last quarter that resulted into a fall in profit? The objective here is to identify the root cause, once identified, the problem can be fixed. So the obvious question that comes into the mind of the decision maker is *what is special about the last quarter?* Before jumping directly into the last quarter to come up with hasty outcomes, the careful decision maker wants to look at the sales data of the last year, but with a different perspective, therefore, his/her question is translated into two queries i.e. (i) What was the quarterly sales at regional level during last year? (ii) What was the quarterly sales at product level during last year? There could be another sequence of questions i.e. probing deeper into the "lossy" quarter directly, which is followed in the next sequence of queries.

It may seem that the decision maker was overly careful, but he was not. Recognize that the benefit of looking at a whole year of data enables the decision maker to identify if there was a trend of sales going down, that resulted in least sales during the last quarter or the sales only went down during the last quarter? The finding was that the sales went down only during the last quarter, so the decision maker probed further by looking at the sales at monthly basis during the last quarter with reference to product, as well as with reference to region. Here is a finding, the sales were doing all right with reference to the product, but they were found to be down with reference to the region, specifically the Northern region.

Now the decision maker would like to know what happened in the Northern region in the last quarter with reference to the products, as just looking at the Northern regional alone would not give the complete picture. Although not explicitly shown in the figure-?, but the current exploratory query of the decision maker is further processed by taking the sales figures of last quarter on monthly basis and coupled together with the product purchased at store level. Now the hard work pays off, as the problem has finally been identified i.e. high cost of products purchased during the last quarter in the Northern region. Well actually it was not hard work in its true literal meaning, as using an OLAP tool, this would have taken not more than a couple of minutes, maybe even less.

Analysis of last example

- Analysis is Ad-hoc
- Analysis is interactive (user driven)
- Analysis is iterative
  - Answer to one question leads to a dozen more
- Analysis is directional
  - Drill Down
  - Roll Up
  - Drill Across

Now if we carefully analyze the last example, some points are very clear, the first one being analysis is ad-hoc i.e. there is no known order to the sequence of questions. The order of the questions actually depends on the outcome of the questions asked, the domain knowledge of the decision maker, and the inference made by the decision maker about the results retrieved. Furthermore, it is not possible to perform such an analysis with a fixed set of queries, as it is not possible to know in advance what questions will be asked, and more importantly what answers or results will be presented corresponding to those questions. In a nut-shell, not only the querying is ad-hoc, but the universal set of pre-programmed queries is also very large, hence it is a very complex problem.

As you would have experienced while using an MIS system or done while developing and MIS system, such systems are programmer driven. One would disagree with that, arguing that development comes after system analysis, when user has given his/her requirements. The problem with decision support systems is that, the user is unaware of their requirements, until they have used the system. Furthermore, the number and type of queries are fixed as set by the programmer, thus if the decision maker has to use the OLTP system for decision making, he is forced to use only those queries that have been pre-written thus the process of decision making is driven not by the decision maker. As we have seen, the decision making is interactive i.e. there is noting in between the decision support system and the decision maker.

In a traditional MIS system, there is an almost linear sequence of queries i.e. one query followed by another, with a limited number of queries to go back to with almost fixed and rigid results. As a consequence, running the same set of queries in the same order will not result in something new. As we discussed in the example, the decision support processes is exploratory, and depending on the results obtained, the queries are further fine-tuned or the data is probed deeply, thus it is an iterative process. Going through the loop of queries as per the requirement of the decision maker fine tunes the results, as a result the process actually results in coming up with the answers to the questions.

Challenges

- Not feasible to write predefined queries.
    - Fails to remain user_driven (becomes programmer driven).
    - Fails to remain ad_hoc and hence is not interactive.
- Enable ad-hoc query support
    - Business user can not build his/her own queries (does not know SQL, should not know it).
    - On_the_go SQL generation and execution too slow.

Now that we have identified how decision making proceeds, this could lure someone with an OLTP or MIS mindset to approach decision support using the traditional tools and techniques. Let's look at this (wrong) approach one point at a time. We identified one very important point that decision support requires much more and many more queries as compared to an MIS system. But even if more queries are written, the queries still remain predefined, and the process of decision making still continues to be driven by the programmer (who is not a decision maker) instead of the decision maker. When something is programmed, it kind of gets hard-wired i.e. it can not be changed significantly, although minor changes are possible by changing the parameter values. That's where the fault lies i.e. hard-wiring. Decision making has no particular order associated with it.

One could naively say OK if the decision making process has to be user driven, let the end user or business user do the query development himself/herself. This is a very unrealistic assumption, as the business user is not a programmer, does not know SQL, and does not need to know SQL. An apparently realistic solution would be to let the SQL generation be on-the-go. This approach does make the system user driven without having the user to write SQL, but even if on-the-go SQL generation can be achieved, the data sets used in a data warehouse are so large, that it is no way possible for the system to remain interactive!

Challenges

- Contradiction
  - Want to compute answers in advance, but don't know the questions
- Solution
  - Compute answers to “all” possible “queries”. But how?
  - NOTE: Queries are multidimensional aggregates at some level

So what are the challenges? We have a paradox or a contradiction here. OK we manage to assemble a very large team of dedicated programmers, with an excellent project manager and all sit and plan to write all queries in advance. Can we do that? The answer is a resounding NO. The reason being, since even the user dose not knows the queries in advance, and consequently the programmers also do not know the queries in advance, so how can the queries be written in advance? It is just no possible.

So what is the solution then? We need a paradigm shift from the way we have been thinking. In a decision support environment, we are looking at the big picture; we are interested in all sales figures across a time interval or at a particular time interval, but not for a particular person. So the solution lies in computing all possible aggregates, which would then correspond to all possible queries. Over here a query is a multidimensional aggregate at a certain level of the hierarchy of the business.

To understand this concept, let’s look at Figure -10.2 that shows multiple levels or hierarchies of geographies. Pakistan is logically divided into provinces, and each province is administratively divided into divisions, each division is divided into districts and within districts are cities. Consider a chain store, with outlets in major cities of Pakistan, with large cities such as Karachi and Lahore having multiple outlets, which you may have observed also. Now the sales actually take place at the business outlet which could be in a zone, such as Defense and Gulberg in our example, of course there may be many more stores in a city, as already discussed.

"All" possible queries (level aggregates)

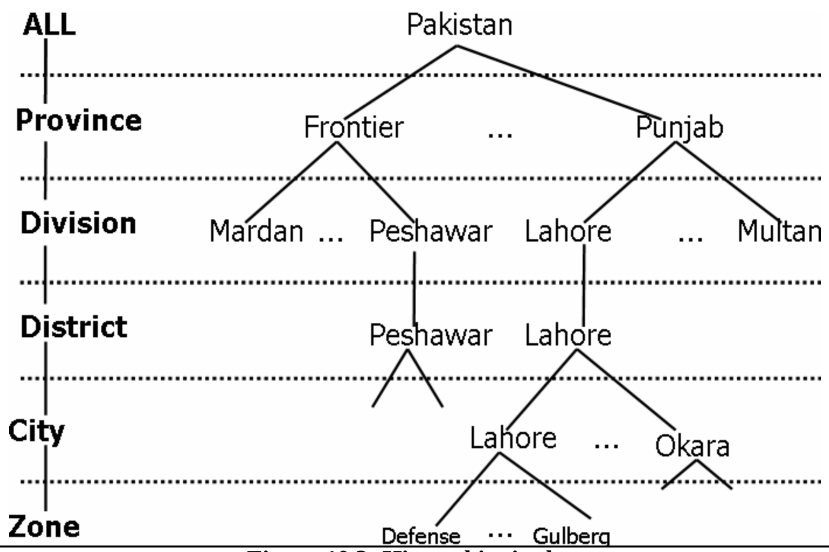


Figure-10.2: Hierarchies in data

OLAP: Facts & Dimensions

- FACTS: Quantitative values (numbers) or "measures."
  - e.g., units sold, sales $, Co, Kg etc.
- DIMENSIONS: Descriptive categories.
  - e.g., time, geography, product etc.
  - DIM often organized in hierarchies representing levels of detail in the data (e.g., week, month, quarter, year, decade etc.).

The foundation for design in this environment is through use of dimensional modeling techniques which focus on the concepts of "facts" and "dimensions" for organizing data. Facts are the quantities, numbers that can be aggregated (e.g., sales $, units sold etc.) that we measure and dimensions are how we filter/report on the quantities (e.g., by geography, product, date, etc.). We will discuss in detail, actually have number of lectures on dimensional modeling techniques.

Where Does OLAP Fit In?

- It is a classification of applications, NOT a database design technique.
- Analytical processing uses multi-level aggregates, instead of record level access.
- Objective is to support very
- (i) fast
- (ii) iterative and
- (iii) ad-hoc decision-making.

OLAP is a characterization of applications. It is NOT a database design technique. People often confuse OLAP with specific physical design techniques or data structure. This is a mistake. OLAP is a characterization of the application domain centered around slice-and-dice and drill down analysis. As we will see, there are many possible implementations capable of delivering OLAP characteristics. Depending on data size, performance requirements, cost constraints, etc. the specific implementation technique will vary.

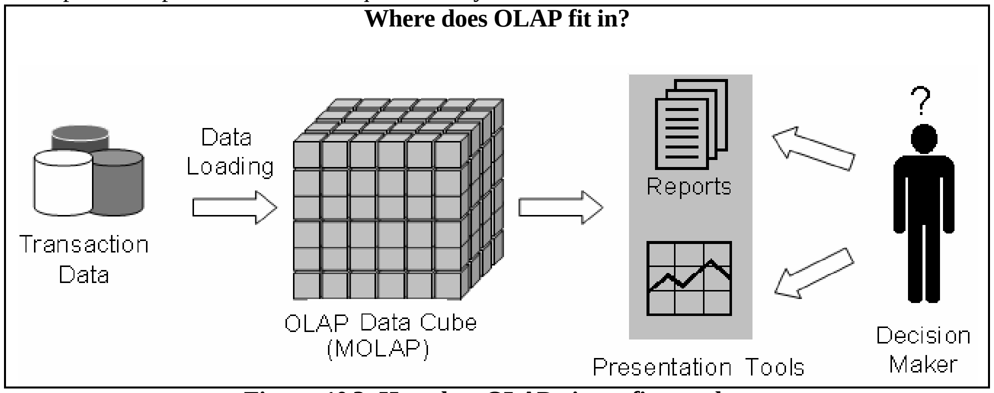


Figure-10.3: How does OLAP pi eces fir together

Figure-10.3 shows that using the transactional databases (after Extract Transform Load) OLAP data cubes are generated. Over here multidimensional cube is shown i.e. a MOLAP cube. Data retrieved as a consequence of exploring the MOLAP cub e is then used as a source to generate reports or charts etc. Actually in typical MOLAP environments, the results are displayed as tables, along with different types of charts, such as histogram, pie etc.

OLTP vs. OLAP

Feature	OLTP	OLAP
Level of data	Detailed	Aggregated
Amount of data retrieved per transaction	Small	Large
Views	Pre-defined	User-defined
Typical write operation	Update, insert, delete	Bulk insert, almost no deletion
“age” of data	Current (60-90 days)	Historical 5-10 years and also current
Tables	Flat tables	Multi-Dimensional tables
Number of users	Large	Low-Med
Data availability	High (24 hrs, 7 days)	Low-Med
Database size	Med (GB- TB)	High (TB – PB)
Query Optimizing	Requires experience	Already “optimized”

Table-10.1: Difference between OLTP and OLAP

The table summarizes the fundamental differences between traditional OLTP systems and typical OLAP applications. In OLTP operations the user changes the database via transactions on detailed data. A typical transaction in a banking environment may transfer money from one account to another account. In OLAP applications the typical user is an analyst who is interested in selecting data needed for decision support. He/She is primarily not interested in detailed data, but usually in aggregated data over large sets of data as it gives the big picture. A typical OLAP query is to find the average amount of money drawn from ATM by those customers who are male, and of age between 15 and 25 years from (say) Jinnah Super Market Islamabad after 8 pm. For this kind of query there are no DML operations and the DBMS contents do not change.

OLAP FASMI Test

Fast: Delivers information to the user at a fairly constant rate i.e. O(1) time. Most queries answered in under 5 seconds.

Analysis: Performs basic numerical and statistical analysis of the data, pre-defined by an application developer or defined ad-hocly by the user.

Shared: Implements the security requirements necessary for sharing potentially confidential data across a large user population.

Multi-dimensional: The essential characteristic of OLAP.

Information: Accesses all the data and information necessary and relevant for the application, wherever it may reside and not limited by volume.

FAST means that the system is targeted to deliver response to moat of the end user queries under 5 seconds, with the simplest analyses taking no more than a second and very few queries taking more than 20 seconds (for various reasons to be discussed). Because if the queries take significantly longer time, the thought process is broken, as users are likely to get distracted, consequently the quality of analysis suffers. This speed is not easy to achieve with large amounts of data, particularly if on-the-go and *ad-hoc* complex calculations are required.

ANALYSIS means that the system is capable of doing any business logic and statistical analysis, which is applicable for the application and to the user, and also keeps it easy enough for the user i.e. at the point -and-click level. It is absolutely necessary to allow the target user to define and execute new *ad hoc* calculations/queries as part of the analysis and to view the results of the data in any desired way/format, without having to do any programming.

SHARED means that the system is not supposed to be a stand-alone system, and should implement all the security requirements for confidentiality (possibly down to cell level) for a multi-user environment. The other point is kind of contradiction of the point discussed earlier i.e. writing back. If multiple DML operations are needed to be performed, concurrent update locking should be executed at an appropriate level. It is true that not all applications require users to write data back, but it is true for a considerable majority.

MULTIDIMENSIONAL is the key requirement to the letter and at the heart of the cube concept of OLAP. The system must provide a multidimensional logical view of the aggregated data, including full support for hierarchies and multiple hierarchies, as this is certainly the most logical way to analyze organizations and businesses. There is no "magical" minimum number of dimensions that must be handled as it is too application specific.

INFORMATION is all of the data and derived information required, wherever it is and however much is relevant for the application. The objective over here is to ascertain the capacity of various products in terms of how much input data they can handle and process, instead of how many Gigabytes of raw data they can store. There are many considerations here, including data duplication, Main Memory required, performance, disk space utilization, integration with data warehouses etc.

OLAP Implementations

1. **MOLAP:** OLAP implemented with a multi-dimensional data structure.

2. **ROLAP:** OLAP implemented with a relational database.

3. **HOLAP:** OLAP implemented as a hybrid of MOLAP and ROLAP.

4. **DOLAP:** OLAP implemented for desktop decision support environments.

MOLAP physically builds "cubes" for direct access - usually in the proprietary file format of a multi-dimensional database (MDD) or a user defined data structure. therefore ANSI SQL is not supported.

ROLAP or a Relational OLAP provides access to information via a relational database using ANSI standard SQL.

HOLAP provides a combination of relational database access and "cube" data structures within a single framework. The goal is to get the best of both MOLAP and ROLAP: scalability (via relational structures) and high performance (via pre-built cubes).

DOLAP allows download of "cube" structures to a desktop platform without the need for shared relational or cube server. This model facilitates a mobile computing paradigm. DOLAP is particularly useful for sales force automation types of applications by supporting extensive slide and dice. A DOLAP implementation needs to be much more efficient in disk and memory utilization than typical server implementations because computing power is often limited on a laptop computer.

MOLAP Implementations

OLAP has historically been implemented using a multidimensional data structure or "cube".

- Dimensions are key business factors for analysis:
  - Geographies (city, district, division, province,...)
  - Products (item, product category, product department,...)
  - Dates (day, week, month, quarter, year,...)

- Very high performance achieved by O(1) time lookup into "cube" data structure to retrieve pre_aggregated results.

Multi-dimensional databases (MDDs) typically use proprietary file formats to store pre-summarized cube structures. There are dozens of vendors who provide MDD products. Essbase from Hyperion and Powerplay from Cognos are some examples of market share leaders in this space. Some RDBMS vendors have also begun packaging MDD capabilities into their product offerings. Express from Oracle (acquired from IRI) and the MOLAP Store in Microsoft's OLAP Server architecture (acquired from Panorama) are examples of products from RDBMS vendors.

MOLAP cubes are built around dimensions, as corresponding to each dimension is the index of the multi-dimensional array data structure. Meaning, if there are four dimensions, then the data will be stored in a four dimensional array i.e. with four indexes i, j, k and l. The other significance of dimension from the business point of view are the key factors they correspond to, typically dimensions are the features on which the WHERE clause is executed. Some examples of dimension are geography, time, products etc. Note that typically dimensions have hierarchies, for example geography dimension has the hierarchy of country, then province, then division, district, city and zone as discussed in the last lecture.

The performance in a MOLAP cube comes from the O(1) look-up time for the array data structure. Recall that to access an array, only the indexes are required i.e. there is no scanning of the array (like a file data structure), there is no hashing it a constant access time operations, similar to a random access memory (or RAM). The only time the time complexity goes beyond O(1) is when the cube size is so large that it can not fit in the main memory, in such a case a page or a block fault will occur.

MOLAP Implementations

- No standard query language for querying MOLAP
  - *No SQL !*

- Vendors provide proprietary languages allowing business users to create queries that involve pivots, drilling down, or rolling up.
  - E.g. MDX of Microsoft
  - Languages generally involve extensive visual (click and drag) support.
  - Application Programming Interface (API)'s also provided for probing the cubes.

As it should be clear by now that, in a MOLAP environment there are no tables, there are no traditional relational structures, hence ANSI SQL can not be used. As a matter of fact, there is no standard query language for querying a MOLAP cube. The cube is sometimes probed by a complex mapping mechanism that maps the user requirements into a set of indices, which are then used to retrieve the aggregate from the corresponding matrix location. However, vendors have come up with proprietary languages for non IT business users, the most popular being MDX by Microsoft. These "languages" are graphical environments where the business users can click on actions and perform drag-and-drop operations to provide input to retrieve data/information from the cube. APIs are also available in the market for programming cube data retrieval by more experienced programmers for running of complex quires. But none of these approaches use SQL.

Aggregations in MOLAP

- Sales volume as a function of (i) product, (ii) time, and (iii) geography
- A cube structure created to handle this.

Figure-11.1: Filled cube

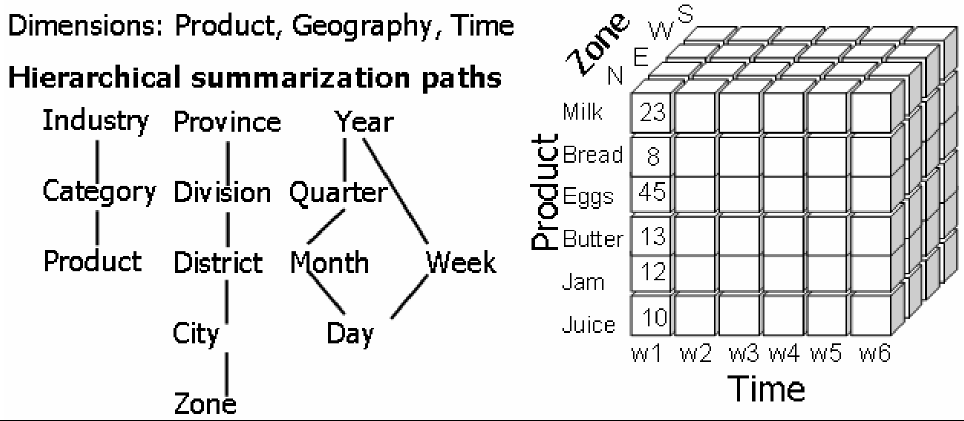


Figure11.1: Aggregation and a MOLAP cube

As already stated a number of times, in a Data Warehouse decision support environment we are interested in the big picture, we want to look at the data from a macro level instead of the micro level. For a macroscopic view aggregates are used. In this example we look at the sales volume i.e. number of items sold as a function of product, time and geography. Note that all three of them are dimensions. The proceed with the analysis, a cube structure will be first created such that each dimension of the cube will correspond to each identified dimension, and within each dimension will be the corresponding hierarchy. The example further shows how the dimensions are "rolled-out" i.e. Province into divisions, then division into district, then district into city and finally cities into zones. Note that weeks could be rolled into a year and at the same time months can be rolled into quarters and quarters rolled into years. Based on these three dimensions a cube is created and shown in figure-11.1.

Cube Operations

- Rollup: summarize data
  - e.g., given sales data, summarize sales for last year by product category and region
- Drill down: get more details
  - e.g., given summarized sales as above, find breakup of sales by city within each region, or within Sindh
- Slice and dice: select and project
  - e.g.: Sales of soft-drinks in Karachi during last quarter
- Pivot: change the view of data

The word cube is synonymously used with a MOLAP, therefore, when we talk of cube operations we are never referring to SQL based querying, instead how we view the data stored in a cube. There are four fundamental cubes operations which are (i) rollup (ii) drill down (iii) slice and dice and (iv) pivoting. In the rollup operation the level of aggregation is increased i.e. data is presented at a higher level of aggregation with less detail. For example while looking at the weekly sales data, the decision maker decides to look at the monthly sales data, this is a rollup operation. On the other hand, while looking at the weekly sales data, the decision makers wants to look at the higher level of detail thus he/she drills down and looks at the daily sales data or was looking at the division level of the sales data, and drills down into district level sales data by looking at the sales in different cities corresponding to a district, among one of the many districts present in a division. Slice and dice is a combination of looking at a subset of data based on more than one dimension. As shown sin the slide, a combination of geography and time is used to look at part of the cube across more than a singe dimension. Finally pivoting is changing the view of the data i.e. replacing the dimension across the (say) x-axis. This will become clear when we look at specific examples in the following slides.

Querying the Cube

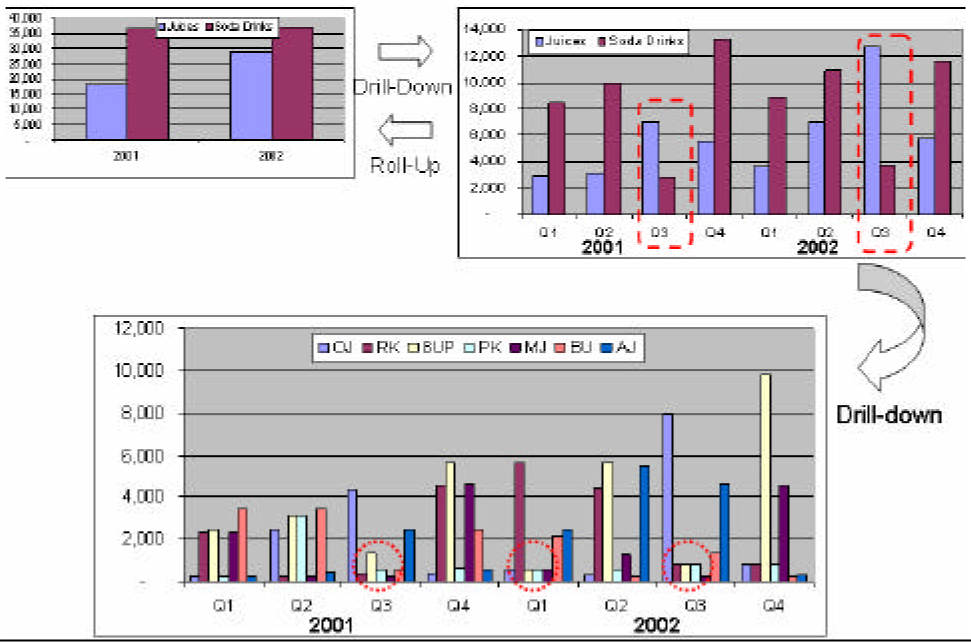


Figure-11.2: Different cube operations

Question: Juices are selling well or the sodas?
Looking at the highest level summary, with the lowest data granularity, shows that the sales of Soda Drinks are higher for years 2001 and 2002.

Question: Which sodas are doing well?
Just clicking on the Sodas column shows the second histogram. It shows that the sales of 8-UP cola are highest in both years i.e. 2001 and 2002.

Question: Is 8-UP a clear winner?
Flipping the dimensions i.e. instead of product on x-axis putting time on x-axis and further drilling-down shows that there is a trend of increasing sales of 8-UP, BUT the sale is alarmingly down during QTR3 in 2001 and QTR1 and QTR3 during 2002.

All of the above information was extracted by exploring 46 table entries and three levels of aggregates. The total exploration time took less than 3 seconds.

Question: Why erratic sales of 8-UP?
This may probably require exploring other dimensions or their combinations.

Querying the Cube: Pivoting

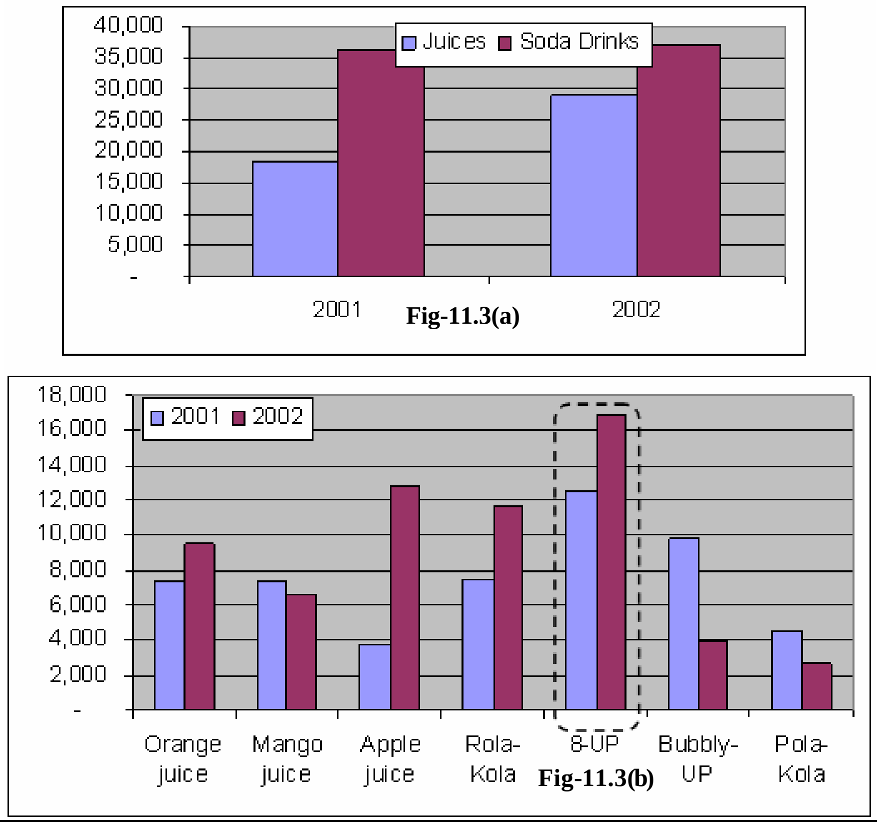


Figure-11.3: Pivot cube operations

This is a good example of pivoting. In the first figure (11.3(a)), along x-axis we have the time dimension, y-axis is the sales volumes, but volumes for the product are under two categories i.e. juices and soda drinks. By the pivoting operation, the dimensions are interchanged, and we end up with having the product along the x-axis and the yearly sales figures (i.e. time dimension) along the y-axis. The purpose of pivoting and other operations are not just to provide a set of different operations, but to enable the decision maker to look at the data from different angles. From fig-11.3(a) it seems that the sales of soda drinks are up consistently through year 2001 and 2002, but after pivoting and doing a drilldown operation, we see that the sales of apple juice was higher than all soda drinks, except 8-UP, but the sale of 8-UP was so high both in years 2001 and 2002, that it shot up the overall sales of soda drinks. Thus all soda drinks are not the winners; actually Pola -Kola has the weakest sales of all.

MOLAP Evaluation

Advantages of MOLAP:

- Instant response (pre-calculated aggregates).
- Impossible to ask question without an answer.
- Value added functions (ranking, % change).

MOLAP implementations with pre-defined cubes as pre-aggregated data perform very well when compared to relational databases, but often have difficulty scaling when the size of dimensions becomes large. The breakpoint is typically around 64,000. Some implementations are also limited to a file size for the cube representation that must be less than 2GB (this is less often an issue than a few years ago).

Actually the cubes are less than 1% full.

MOLAP Evaluation

Drawbacks of MOLAP:

- Long load time (pre -calculating the cube may take days!).
- Very sparse cube (wastage of space) for high cardinality (sometimes in small hundreds). e.g. number of heaters sold in Jacobabad or Sibi.

A MOLAP is in no way a win-win situation, it has its won intrinsic pitfalls, which does not make it an overall winner. The biggest drawback is the extremely long time taken to pre-calculate the cubes, remember that in a MOLAP all possible aggregates are computed. The number of aggregates suffers from something called as the "curse of dimensionality" i.e. as the number of dimensions increases, the number of possible aggregates increases exponentially, this will be further clarified in an upcoming slide. Because of long calculation times, cube creation is a batch process and usually has the lowest priority and scheduled to be done late in the night. This actually turns out to be a big mistake (as we will discuss subsequently) as it may so happen that the cube generation may not actually take place, and the decision makers are presented with the old and stale data, and they tend to lose faith in the OLAP paradigm. Although the number of possible aggregates is very large, but NOT all the aggregates may have values, there can be and will be quite a few aggregates which will have null values. For example, many of the items sold in winter are not sold in summer and not even kept in the store (and vice-a-versa). Consequently, there are no corresponding sales, and if the cube is generated that includes all the items, there will be many null aggregates, resulting in a very sparse cube. This will result in requirement of large amount of memory, most of which would be wasted. For these very reasons cube compression is a hot area of research and development. We will not discuss it any further.

MOLAP Implementation Issues

Maintenance issue: Every data item received must be aggregated into *every* cube (assuming "to-date" summaries are maintained). Lot of work.

Storage issue: As dimensions get less detailed (e.g., year vs. day) cubes get much smaller, but storage consequences for building hundreds of cubes can be significant. Lot of space.

Scalability:
Often have difficulty scaling when the size of dimensions becomes large. The breakpoint is typically around 64,000 cardinality of a dimension.

Maintenance Considerations: Every data item received into MDD must be aggregated into *every* cube (assuming "to-date" summaries are maintained). Have to update the cubes every time a change occurs (insert, delete, update), otherwise the user will run into synchronization problems.

Storage Considerations: Although cubes get much smaller (e.g., more dense) as dimensions get less detailed (e.g., year vs. day), storage implications for building hundreds of cubes can be significant. The reason being, as each possible combination of dimensions has to be pre-calculated i.e. every combination of every aggregate, so soon faced with a combinatorial explosion. Therefore, the number of entries in each dimension has to be of some reasonable number. If you have entries above 100 in each dimension then things tend to blow up. When does it actually blows up depends on the tool being used and how sparse matrices are stored. So you might end up requiring more cube space then allocated or even available. Consider taking each data element by week, and then 65 weeks times' geography is one cube. Now consider taking each data element by month, this is a different cube i.e. by geography by 12 months, and then a different cube by geography by year and so on. So the combinatorial explosion gets way out of hand.

Scalability:
MOLAP implementations with pre-defined cubes as pre-aggregated data perform very well when compared to relational databases, but often have difficulty scaling when the size of dimensions becomes large. The breakpoint is typically around 64,000 cardinality of a dimension. Typically beyond tens (sometimes small hundreds) of thousands of entries in a single dimension will break the MOLAP model because the pre-computed cube model does not work well when the cubes are very sparse in their population of individual cells. Some implementations are also limited to a file size for the cube representation that must be less than 2GB (this is less often an issue than a few years ago). You just can not build cubes big enough, or enough of them to have every thing pre-computed. So you get into the problems of scale. As already discussed, it is difficult to scale because of combinatorial explosion in the number and size of cubes when dimensions of significant cardinality are required. There are two possible, but limited solutions addressing the scalability problem i.e. Virtual cubes and partitioned cubes.

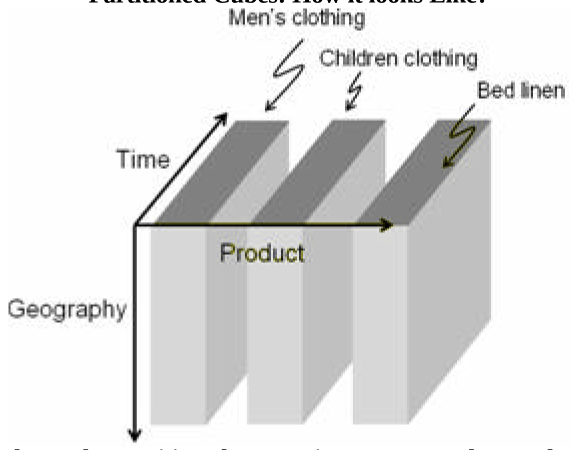
Partitioned Cubes

- To overcome the space limitation of MOLAP, the cube is partitioned.
- One logical cube of data can be spread across multiple physical cubes on separate (or same) servers.
- The divide&conquer cube partitioning approach helps alleviate the scalability limitations of MOLAP implementation.
- Ideal cube partitioning is completely invisible to end users.
- Performance degradation does occur in case of a join across partitioned cubes.

When the cardinality of a dimension forces a cube to become larger than what can reasonably be supported in a single cube, it is common practice to partition the cube into multiple "sub-cube" instantiations. The sub-cubes are usually defined around logical partitions within the dimensional hierarchies. For example, if there are a large number of entries at the most detailed level in a product hierarchy, it may be advisable to create distinct cubes along product category boundaries to divide-and-conquer. Another example would be to take a geographical dimension and partition into cubes whereby each region has its own cube.

Advanced MOLAP implementations (e.g., Microsoft Analytic/OLAP Services) will automatically distribute queries across partitioned cubes (potentially on distinct server platforms) for parallel retrieval without exposing the end user to the underlying physical deployment of the partitioned cubes.

Partitioned Cubes: How it looks Like?



Sales data cube partitioned at a major cotton products sale outlet

Figure-11.4: Partitioned Cube

There is unlikely to be a relationship between the sales of men's and children's clothing, thus the sales cube is partitioned, as shown in figure-11.4. Usually when men buy, they buy for themselves in most cases, except of course when they buying a gift for someone. When women buy, they may also buy for the children, but not men. But then this may be incorrect for a certain culture, hence is not a rule of thumb.

Bed linen is something very different, hence the original single cube of all sales aggregates can be partitioned along these two logical boundaries into three cubes which require less storage space per cube. As MOLAP is intrinsically main memory resident, hence cube partitioning does not results in notable speed-up, but does results in memory requirements, if for most queries one or two cube partitions are required with infrequent joins across them.

Virtual Cubes

Used to query two dissimilar cubes by creating a third "virtual" cube by a join between two cubes.

- Logically similar to a relational view i.e. linking two (or more) cubes along common dimension(s).
- Biggest advantage is saving in space by eliminating storage of redundant information.

Example: Joining the store cube and the list price cube along the product dimension, to calculate the sale price without redundant storage of the sale price data.

Advanced MOLAP implementations will often facilitate the creation of virtual cubes by combining multiple cubes (at run-time) along common dimensions between cubes.

There is a run-time cost to "joining" across cubes at the time of query execution, but the savings in storage can be quite large by eliminating redundancy in the underlying data representation. Typically, the virtual cubes involve "joining" cubes along common dimensions whereby one cube has a (smaller) subset of the dimensions from the other (bigger) cube.

Example: Usually the sale price of different items varies based on the geography and the time of the year. Instead of storing this information using an additional dimension, the said price is calculated at run-time, this may be slow, but can result in tremendous saving in space, as all the items are not sold throughout the year.

Lecture-12: Relational OLAP (ROLAP)

Why ROLAP?

Issue of scalability i.e. curse of dimensionality for MOLAP

- Deployment of significantly large dimension tables as compared to MOLAP using secondary storage.
- Aggregate awareness allows using pre-built summary tables by some front-end tools.
- Star schema designs usually used to facilitate ROLAP querying (in next lecture).

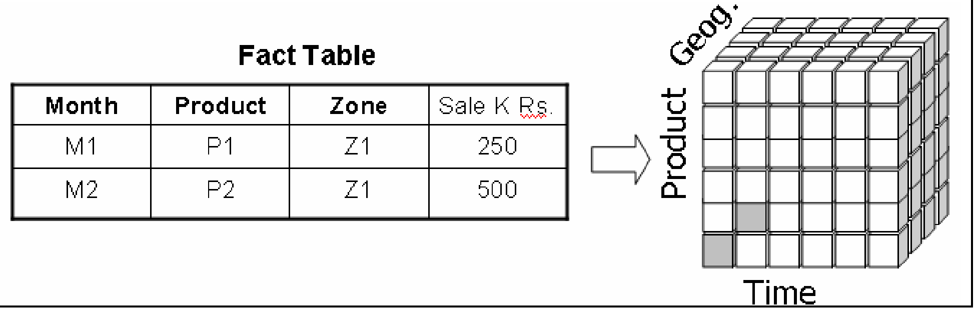
The biggest problem with MOLAP is the requirement of large main memory as the cube size increases. There may be many reasons for the increase in cube size, such as increase in the number of dimensions, or increase in the cardinality of the dimensions, or increase in the amount of detail data or a combination of some or all these aspects. Thus there is an issue of scalability which limits its applications to large data sets.

Despite advances in MOLAP technologies, high-end OLAP implementations will normally require assistance from a relational database. Hence a ROLAP or Relational OLAP. ROLAP tools will query the relational database using SQL generated to conform to a framework using the facts and dimensions paradigm using the star schema.

The other approach is "aggregate awareness" i.e. the environment is smart enough to develop or compute higher level aggregates using lower level or more detailed aggregates.

ROLAP as a "Cube"

- OLAP data is stored in a relational database (e.g. a star schema)
- The fact table is a way of *visualizing as* an "un-rolled" cube.
- So where is the cube?
  - It's a matter of perception
  - Visualize the fact table as an elementary cube.



Fact Table

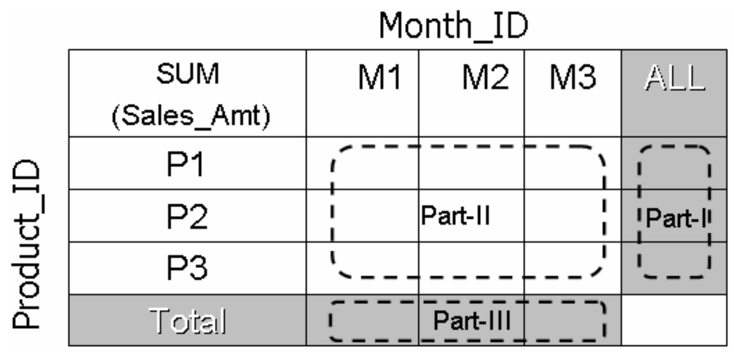
Month	Product	Zone	Sale K Rs.
M1	P1	Z1	250
M2	P2	Z1	500

Figure-12.1: ROLAP as a cube

Recall from the last lecture that an implementation of the OLAP framework will be accepted as OLAP after it passes the FAS**M**I test i.e. it has to be **M**ulti-dimensional. The question is "how to convert something into a cube" which is intrinsically not a cube. And more importantly, if MOLAP has the space requirement limitations, and to overcome those limitations we are using a different implementation, then wouldn't the conversion back to a "cube" defeat the purpose? This is actually a very good question, and needs some detailed explanation. Fig-12.1 shows two rows of a fact table, and the aggregates corresponding to the cells of the fact table are shown correspondingly in a three dimensional cube. Extending this argument, each and every cell of the fact table can have a corresponding mapping. To clarify this further, consider the analogy of the map of the world. Our planet earth is NOT flat, it is spherical in nature i.e. three dimensional, yet the map of the earth is flat i.e. two dimensional in nature i.e. there is a mapping from 3D space to 2D space. If you look closely at the map, you will see that the map is divided into a grid based on longitude and latitude, and the corresponding cells or rectangles are not of the same size. Similarly a 2D table can be considered to be a mapping or representation of a multi-dimensional cube or vice-a-versa.

How to create a "Cube" in ROLAP

- Cube is a logical entity containing values of a certain fact at a certain aggregation level at an intersection of a combination of dimensions.
- The following table can be created using **3** queries



SUM (Sales_Amt)	M1	M2	M3	ALL
P1				
P2		Part-II		Part-I
P3				
Total		Part-III		

Month_ID
Product_ID

Figure-12.2: Creating tables from queries

When we talked of a cube for a MOLAP, it was not actually a physical cube, but was a logical entity. We continue with that concept, and assume that what was stored in a cube at a certain combination of indexes, corresponding to such a group of indices, we store the aggregates in a two dimensional table, and we use such groups of tables to store the same data that was stored in a MOLAP. The table shown in fig-12.2 is divided into three parts shown shaded and also shown by dotted lines. Corresponding to each part of the table, there is a query and consequently the table can actually be filled using three SQL queries as follows:

- For the table entries, without the totals

```
SELECT      S.Month_Id, S.Product_Id,     SUM(S.Sales_Amt)
```

```
FROM        Sales
GROUP BY    S.Month_Id, S.Product_Id;
```

- For the row totals

```
SELECT      S.Product_Id, SUM (Sales_Amt)
FROM        Sales
GROUP BY    S.Product_Id;
```

- For the column totals

```
SELECT      S.Month_Id, SUM (Sales)
FROM        Sales
GROUP BY    S.Month_Id;
```

The first query can be used to fill Part-II of the table, the second query used to fill Part-I of the table, and the third query used to fill Part -III of the table, thus using these three queries, we create a ROALP structure.

Problem with simple approach

- Number of required queries increases exponentially with the increase in number of dimensions.
    - It's wasteful to compute all queries.
    - In the example, the first query can do most of the work of the other two queries
    - If we could save that result and aggregate over Month_Id and Product_Id, we could compute the other queries more efficiently

Using typical SQL to fill-up the tables quickly runs into a problem, as the number of dimensions increases, the number of aggregates also increases, and the number of queries required to calculate those aggregates also increases. Actually it becomes extremely wasteful to compute all queries, wasteful, because if we are smart, we can use the results of the queries already computed to get the answers to new queries. How to do this? it is not very difficult. For example for the column total queries, we could just add the aggregates over the results of the months. So the moral of the story is "Work smart not hard".

Cube clause

- The CUBE clause is part of SQL:1999
    - GROUP BY CUBE (v1, v2, …, vn)
    - Equivalent to a collection of GROUP BYs, one for each of the subsets of v1, v2, …, vn

The other problem with using standard SQL is that one has to write too many statements and that could lead to mistakes. Therefore, back in 1999 a CUBE clause was made part of SQL, and that clause is equivalent to a collection of GROUP BY clauses.

Some students who did a BS final year project with me of an HOLAP implementation, used dynamic web page generation to dynamically generate SQL instead of hard-coding the queries to generate the aggregates. Meaning, they used SQL to generate aggregates to fill a MOLAP cube. The project was a success, all got jobs based on this work; the first prize in the 13th Annual National Software Competition along with a cash prize of Rs. 30,000 was a bonus.

ROLAP and Space Requirement

If one is not careful, with the increase in number of dimensions, the number of summary tables gets very large

Consider the example discussed earlier with the following two dimensions on the fact table...

Time: Day, Week, Month, Quarter, Year, All Days

Product: Item, Sub-Category, Category, All Products

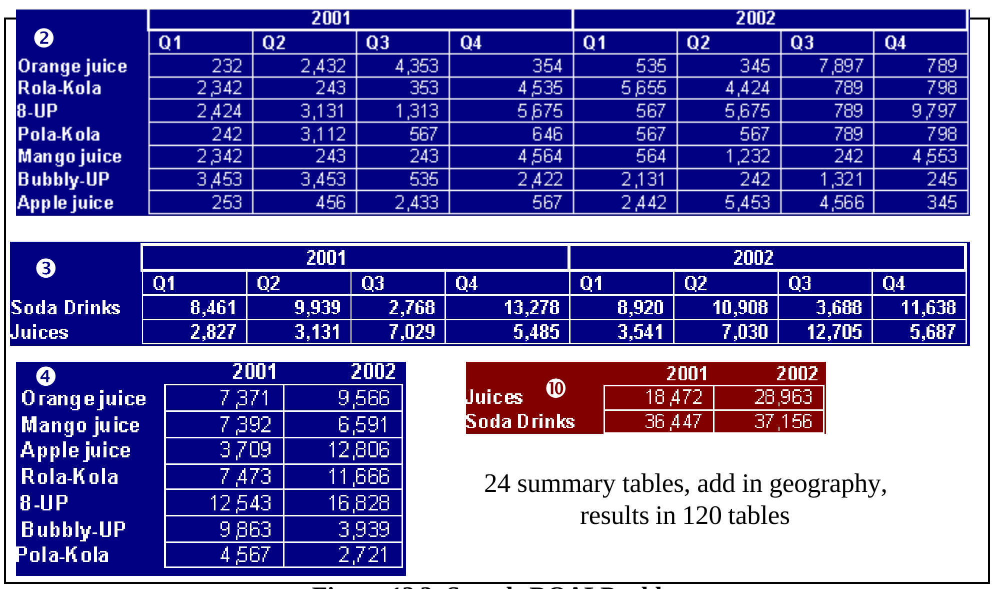
OK so we worked smart and got around the problem of aggregate generation. But the aggregates once generated have to be stored somewhere too i.e. in tables as this is a ROLAP environment.

Be warned: Pre-aggregates can very quickly get out of control in a ROLAP environment. Do not try to pre-aggregate all combinations of all dimensions at all levels of the hierarchies. If you do, the storage and maintenance costs will quickly overwhelm your implementation.

For example, consider the combinatorial explosion with just two dimensions as shown above...

EXAMPLE: ROLAP & Space Requirement

A naïve implementation will require all combinations of summary tables at each and every aggregation level.



❷	2001				2002			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Orange juice	232	2,432	4,353	354	535	345	7,897	789
Rola-Kola	2,342	243	353	4,535	5,655	4,424	789	798
8-UP	2,424	3,131	1,313	5,675	567	5,675	789	9,797
Pola-Kola	242	3,112	567	646	567	567	789	798
Mango juice	2,342	243	243	4,564	564	1,232	242	4,553
Bubbly-UP	3,453	3,453	535	2,422	2,131	242	1,321	245
Apple juice	253	456	2,433	567	2,442	5,453	4,566	345

❸	2001				2002			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Soda Drinks	8,461	9,939	2,768	13,278	8,920	10,908	3,688	11,638
Juices	2,827	3,131	7,029	5,485	3,541	7,030	12,705	5,687

❹	2001	2002
Orange juice	7,371	9,566
Mango juice	7,392	6,591
Apple juice	3,709	12,806
Rola-Kola	7,473	11,666
8-UP	12,543	16,828
Bubbly-UP	9,863	3,939
Pola-Kola	4,567	2,721

❿	2001	2002
Juices	18,472	28,963
Soda Drinks	36,447	37,156

24 summary tables, add in geography, results in 120 tables

Figure-12.3: Sample ROALP tables

There are 24 possible pre-aggregates just with the product and Time dimensions! Add in the geography dimension and we are quickly up to 120 pre-aggregates (and the number of levels in the hierarchies that we are assuming is very conservative). The largest aggregate will be the summarization by day and product because this is the most detailed. Clearly, we do not want to pre-aggregate all of these combinations.

Smart tools will allow less detailed aggregates to be constructed from more detailed aggregates (full aggregate awareness) at run-time so that we do not go all the way down to the detail for every aggregation. However, for this to work, the metrics must be additive (e.g., no ratios, averages, etc.). More detailed pre-aggregates are larger, but can also be used to build less detailed aggregates on-the-go.

ROLAP Issues

- Maintenance.
- Non standard hierarchy of dimensions.
- Non standard conventions.
- Explosion of storage space requirement.
- Aggregation pit-falls.

Creating an aggregate is relatively easy as compared to keeping it updated, the maintenance issue will eat you alive, if you (say) you get data late, and it is also to be reflected in the aggregate archives.
Dimensions are not just simply atomic item, then SKU, then product category. They can cross boundaries too and can become expensive to create.

The conventions for (say) week may be absolute within an organization, but differ across organizations, worst they could differ across the organization too e.g. marketing people looking differently at the week as compared to the accounts people.

ROLAP Issue: Maintenance

Summary tables are mostly a maintenance issue (similar to MOLAP) than a storage issue.

- Notice that summary tables get much smaller as dimensions get less detailed (e.g., year vs. day).
- Should plan for twice the size of the un-summarized data for ROLAP summaries in most environments.
- Assuming "to-date" summaries, every detail record that is received into warehouse must aggregate into EVERY summary table.

It is not unusual for the collection of pre-aggregate summary tables to take up significantly more space (e.g., double or higher) than the raw data tables in a deployment that is focused heavily on query performance delivery. However, the maintenance cost for keeping all of the pre-aggregates up-to-date with the detailed data (very important for consistent query results) is very high. Pre-aggregate summary tables in the relational environment have the same "update problem" as cubes in a MOLAP environment.

ROLAP Issue: Hierarchies

Dimensions are NOT always simple hierarchies

Dimensions can be more than simple hierarchies i.e. item, subcategory, category, etc.

The product dimension might also branch off by trade style that cross simple hierarchy boundaries such as:

Looking at sales of air conditioners that cross manufacturer boundaries, such as COY1, COY2, COY3 etc.

Looking at sales of all "green colored" items that even cross product categories (washing machine, refrigerator, split-AC, etc.).

Looking at a combination of both.

It is quite common that dimensions are more than simple hierarchies. A simple product hierarchy might include item, subcategory, category, etc. However, the product dimension might also branch off by trade style in ways that cross boundaries of the simple hierarchy. We may want to look at sales of air conditioners that cross manufacturer boundaries, such as COY1, COY2, COY3 etc. Looking at all "green colored" items will cross product categories (washing machine, refrigerator, split-AC, etc.). Looking at a combination of both, or other will result in a combinatorial explosion, as these combinations get very large - such that brute-force pre-aggregation is not practical.

Providing analytic capability along multip le calendar hierarchies is even more unmanageable in terms of pre-aggregate construction.

ROLAP Issue: Semantics

Conventions are NOT absolute
Example: What is calendar year? What is a week?

- Calendar:

01 Jan. to 31 Dec or
01 Jul. to 30 Jun. or
01 Sep to 30 Aug.

- Week:

Mon. to Sat. or Thu. to Wed.

Conventions may vary across organizations, and even within an organization. Consider the apparently simple case of calendar year. There is no absolute standard definition of calendar year i.e. is it Jan to Dec. or Jun to Jul or any other time period covering 12 months? Similarly what is meant by week i.e. it starts from which day and ends at which day? Even within an organization the nomenclature may vary, for example finance people may consider a Mon. to Sun. week, while the marketing people may consider a Wed. to Tue. Week. Consequently the results of aggregation will vary across the organization for the same data, thus creating lot of confusion. To bring all the departments on the same grid may require lot of arm twisting from the highest level and will involve office politics too.

ROLAP Issue: Storage Space Explosion

Summary tables required for non-standard grouping

Summary tables required along different definitions of year, week etc.

Brute force approach would quickly overwhelm the system storage capacity due to a combinatorial explosion.

We just saw the problem of semantics and thought it was complex, we were wrong! There are many more and other non-standard grouping too. For example during the month of August to capitalize on the patriotism because of the Independence Day, different manufacturers print the national flag on the packages. This adds a new grouping, as the decision maker would be interested in knowing the sales of item with flag as compared to items without a flag printed on them.

In some cases bringing all the departments on the same grid for agreeing on the same definition of year or week may not be advisable, in such a case the number of summary tables increases, as tables are required for each definition of the week and the year. One may naively follow the brute force path i.e. creating all possible summary tables with all possible definitions, groupings and nomenclatures. This may work for small databases, but for VLDB (Very Large Data Bases) very soon the memory requirements of the system will be chocked. Thus this is not a viable option for large data sets.

ROLAP Issue: Aggregation Pitfalls

- Coarser granularity correspondingly decreases potential cardinality.
- Aggregating whatever that can be aggregated.
- Throwing away the detail data after aggregation.

As data is aggregated to higher levels in the dimensional hierarchy the number of rows retained will obviously reduce. However, this reduction is not usually directly proportional to the ratio of entries at each level in the hierarchy. For example, one might (incorrectly) think that the weekly summary table will be about 14% or 1/7th of the daily summary table as there are seven days in a week. Therefore, keeping the data at a daily level of granularity should be seven times the size of the weekly level, right? Wrong. The reason being all products that sell in a week definitely do not sell every day of the week. As a result, the factor is 2× rather than 7×. For example rent al of movies usually goes up during the weekends. Most heart patients suffer a stroke on Monday, so are most of stock sales.

A general rule of thumb is that aggregation from one level of detail to another should only be undertaken if there is a 10x (or more) reduction in table size from the more detailed level to the coarser level of aggregation. Of course, this is only a guideline and will be influenced by the frequency of query activity at each level of potential aggregation.

Keeping the detail is impo rtant because it is inevitable that advanced analysis will require drill down to the most granular level of detail available. Remember Murphy's Law, the day you throw away the detail, is the day it would be required.

How to reduce summary tables?

Many ROLAP products have developed means to reduce the number of summary tables by:

- Building summaries on-the-fly as required by end-user applications.
- Enhancing performance on common queries at coarser granularities.
- Providing smart tools to assist DBAs in selecting the "best" aggregations to build i.e. trade-off between speed and space.

Tools from vendors such as Microsoft and Micro Strategy are “fully aggregate aware” with the ability to take summaries from one (more detailed) level of aggregation and roll them up into a less detailed summary at run-time. In this way, aggregate metrics can be delivered without forcing run-time aggregation from the most detailed data in the warehouse or mart.

Wizards have come into the marketplace with the ability to make suggestions as to optimal aggregates that should be built for maximizing performance for a defined workload.

Performance vs. Space Trade-off

- Maximum performance boost implies using lots of disk space for storing every pre-calculation.
- Minimum performance boost implies no disk space with zero pre-calculation.
- Using Meta data to determine best level of pre-aggregation from which all other aggregates can be computed.

Theoretically there can be two extremes i.e. free space and free performance. If storage is not an issue, then just pre-compute every cube at every unique combination of dimensions at every level as it does not cost anything. This will result in maximum query performance. But in reality, this implies huge cost in disk space and the time for constructing the pre-aggregates. In the other case where performance is free i.e. infinitely fast machines and infinite number of them, then there is not need to build any summaries. Meaning zero cube space and zero pre-calculations, and in reality this would result in minimum performance boost, in the presence of infinite performance.

What is meant by Meta data? Meta data is data about data. So what is the data about data one would be interested in? For example, how many entries are there at each level of the hierarchy? How many stores are there? How many stores per zone? How many zones per district? Etc. When the density of each dimension is known, it gives a fair idea where the aggregation is going to have the biggest bang for the buck. Because if I have a dimension that has for example on average two UPCs (Universal Product Code) per SKU (Stock Keeping Unit), its really not very interesting to build a summary of UPCs in SKUs because all it saves is adding two records together.

Performance vs. Space Trade-off using Wizard

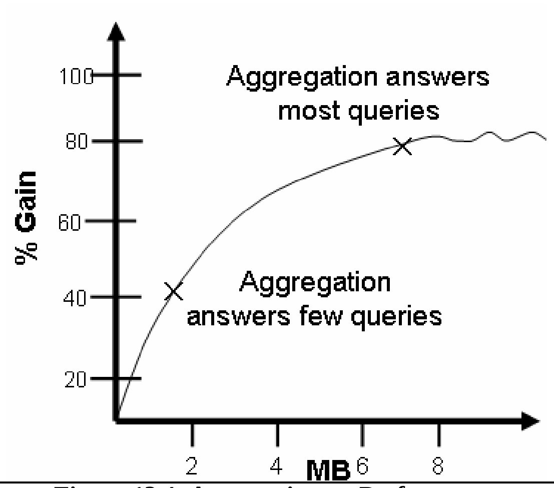


Figure-12.4: Aggregation vs. Performance

Aggregation design wizards allow a cube or pre-aggregate designer to specify the tradeoff between disk storage and performance to determine the maximum volume of pre-calculated aggregates, as shown in Figure-12.4. Suggestions about amount of aggregation are based on the amount of data reduction along different dimensions and the size of each aggregate.

Optimizing the amount of aggregation is a very hard problem, thus heuristics are used. Heuristics in the aggregate design wizard can be improved by Usage-Based Optimization Wizards via examination of query logs to determine which pre-aggregate will deliver best bang for your storage buck.

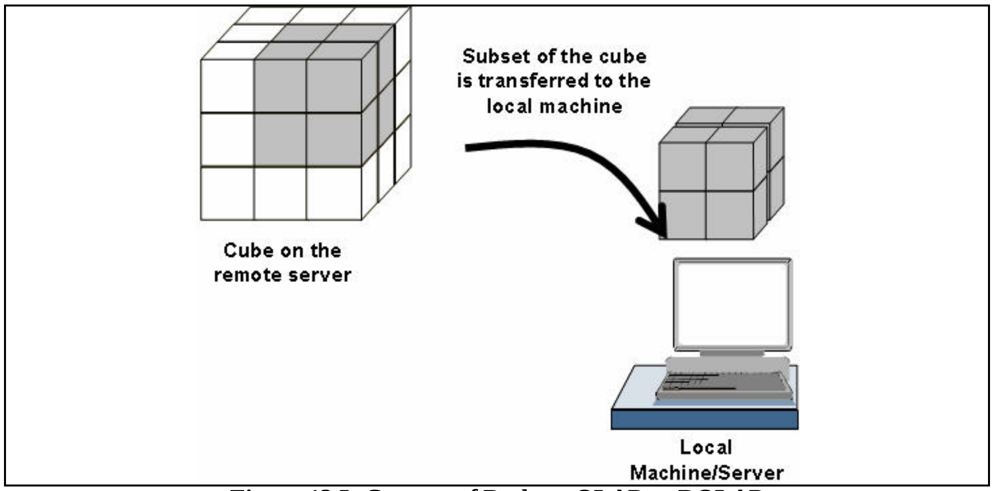
Microsoft's Usage-Based Optimization Wizard (shown here) also allows the DBA to tell Analytic (OLAP) Services to create a new set of aggregations for all queries exceeding a defined response time threshold.

HOLAP

- Target is to get the best of both worlds.
- HOLAP (Hybrid OLAP) allow co-existence of pre-built MOLAP cubes alongside relational OLAP or ROLAP structures.
- How much to pre-build?

The hybrid OLAP (HOLAP) solution is a mix of MOLAP and relational ROLAP architectures that supports queries against summary and transaction data in an integrated fashion. HOLAP environments use MOLAP cubes to support common access paths with reasonably small dimensional cardinality and number of dimensions and relational structures when greater scalability for OLAP queries is required. This coexistence strategy allows exploiting the best of both worlds. Microsoft OLAP Services supports a HOLAP environment, as do tools such as HOLOS. The HOLAP approach enables a user to perform multidimensional analysis on data in the MDDB along with query based probing. However, if the user reaches the bottom of the multidimensional hierarchy and requires further detail data, the smart HOLAP engine automatically generates SQL to retrieve the detail data from the source RDBMS and returns it to the end user. This is done transparently to the user. Several MOLAP vendors, such as Arbor and Oracle, have transitioned to HOLAP architectures that include a ROLAP component. However, these HOLAP architectures are typically more complex to implement and administer than ROLAP or MOLAP architectures individually.

DOLAP


Subset of the cube is transferred to the local machine
Cube on the remote server
Local Machine/Server
Figure-12.5: Concept of Desktop OLAP or DOLAP

DOLAP typically is the simplified version of MOLAP or ROLAP. DOLAP is inexpensive, it is fast and easy to setup on small data sets comprising of thousands of rows instead of millions of rows. It provides specific cube for the analysis. The DOLAP systems developed are extensions of production system report writers, while the systems developed in the early days of client /server computing aimed to take advantage of the power of the emerging PC desktop machine. DOLAP also provides the mobile operations of OLAP for the people who travel and move extensively, such as sales people. The one obvious disadvantage of DOLAP is that it lacks the ability to manage large data sets. But this is just another technique to suit the business requirement.

Lecture-13
Dimensional Modeling (DM)

Relational modeling techniques are used to develop OLTP systems. A typical OLTP system involves a very large number of focused queries i.e. accesses a small number of database rows, which are accessed mostly via unique or primary keys using indexes. The relational model, with well thought-out de-normalization, can result in a physical database design that meets the requirements of OLTP systems. Decision support systems, however, involve fundamentally different queries. These systems involve fewer queries that are usually not focused i.e. access large percentages of the database tables, often performing joins on multiple tables and in many cases aggregating and sorting the results. OLTP systems are NOT good at performing joins.

In the early days of DSS, OLTP systems were used for decision support with very large databases and performance problems were encountered. Some surprising findings were the deterioration of the performance of the DBMS optimizers when the number of tables joined exceeded a certain threshold. This problem led to development of a second type of model that "flattened out" the relational model structures by creating consolidated reference tables from several pre-joined existing tables i.e. converting snow-flakes into stars.

However, according to one school of thought, DM should not be the first reaction, but followed after trying indexing, and views and weighing different options carefully.

The need for ER modeling?

- Problems with early COBOLian data processing systems.
- Data redundancies.
- From flat file to *Table*, each entity ultimately becomes a *Table* in the physical schema.
- Simple $O(n^2)$ Join to work with *Tables.*

ER is a logical design technique that seeks to remove the redundancy in data. Imagine the WAPDA line-man going from home to home and recording the meter reading from each customer. In the early COBOLian days of computing (i.e. long before relational databases were invented), this data was transferred into the computer from the original hand recordings as a single file consisting of number of fields, such as name of customer, address, meter number, date, time, current reading etc. Such a record could easily have been 500 bytes distributed across 10 fields. Other than the recording date and meter reading, EVERY other field was getting repeated. Although having all of this redundant data in the computer was very useful, but had its down sides too, from the aspect of storing and manipulating data. For example, data in this form was difficult to keep consistent because each record stood on its own. The customer's name and address appeared many times, because this d ata was repeated whenever a new reading was taken. Inconsistencies in the data went unchecked, because all of the instances of the customer address were independent, and updating the customer's address was a chaotic transaction as there are literally a dozen ways to just represent “house no”!

Therefore, in the early days of COBOLian computing, it became evident (I guess first to Dr. Codd) to separate out the redundant (or repeating) data into distinct tables, such as a customer master table etc., but at a price. The then available systems for data retrieval and manipulation became overly complex and inefficient because, this required careful attention to the processing algorithms for linking these sets of tables together i.e. join operation. Note that a typical nested loop join will run in $O(n^2)$ time if the two tables are separate such as customer and product bought by the customers. This requirement resulted in database system that was very good at linking tables, and paved the way for the relational database revolution.

Why ER Modeling has been so successful?

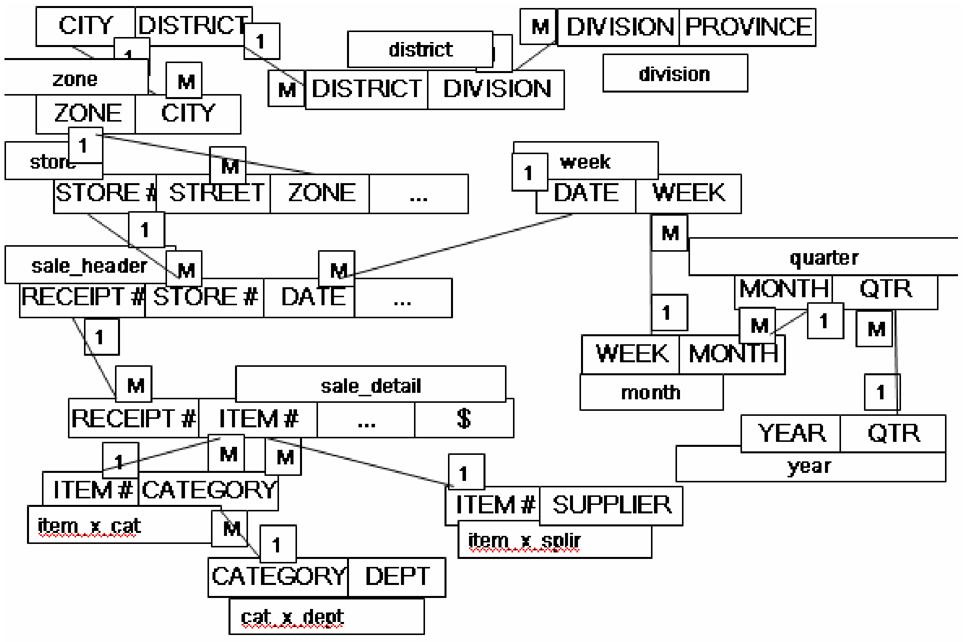
- Coupled with normalization drives out all the redundancy from the database.
- Change (or add or delete) the data at just one point.
- Can be used with indexing for very fast access.
- Resulted in success of OLTP systems.

The ER modeling technique is a discipline used to highlight the microscopic relationships among data elements or entities. The pinnacle of achievement of ER modeling is to remove all redundancy from the data. This is enormously beneficial for the OLTP systems, because transactions are made very simple and deterministic i.e. no surprises in the overall query execution time i.e. it must finish (say) under 5 seconds. The transaction for updating customer's address gets reduced to a single record lookup in a customer address master table. This lookup is controlled by a customer address key, which defines uniqueness of the customer address record i.e. PK. This lookup is implemented using an index, hence is extremely fast. It can be stated without hesitation, that the ER techniques have contributed to the success of the OLTP systems.

Need for DM: Un-answered Qs

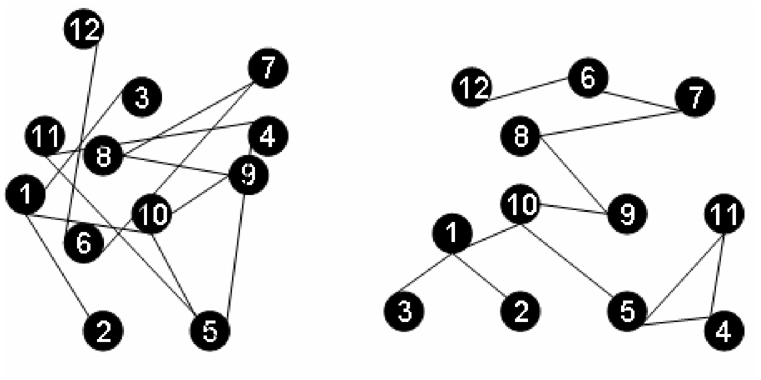
- Lets have a look at a typical ER data model first.
- Some Observations:
  - All tables look-alike, as a consequence it is difficult to identify:
    - Which table is more important ?
    - Which is the largest?
    - Which tables contain numerical measurements of the business?
    - Which table contain nearly static descriptive attributes?

Is DM really needed? In order to better understand the need for DM lets have a look at the diagram showing the retail data in simplified 3NF.



Having a close look at the diagram, it reveals that a separate table has been maintained to store different entities resulting in a complex overall model. Probing the model to get informative insight about data is not straightforward. For example, we can not tell by looking at the model the relative importance of tables. We can not know the table sizes (though we can guess by looking at just headers), and most importantly we can not tell about business dimensions by just looking at the table headers i.e. which tables contain business measurements of the business , which tables contain the static descriptive data and so on.

- Many topologies for the same ER diagram, all appearing different.
  - Very hard to visualize and remember.



  - A large number of possible connections to any two (or more) tables

Figure-13.1: Need for DM: Un-answered Qs

Even for the simple retail example, there are more than a dozen tables that are linked together by puzzling spaghetti of links; and this is just the beginning. This problem is further complicated by the fact that there can be numerous (actually exponentially large) topologies for the same system created by different people, and re-orienting makes them better (or worse).

The problem becomes even more complex for the ER model for an enterprise, which has hundreds of logical entities, and for a high end ERP systems for a large multinational there could be literally thousands of such entities. Each of these entities typically translates into a physical table when the database is implemented. Making sense of this "mess" is almost impossible, communicating it to someone even more difficult.

Need for DM: The Paradox

- **The Paradox:** Trying to make information accessible using tables resulted in an inability to query it efficiently!
- ER and Normalization result in large number of tables which are:
  - Hard to understand by the users (DB programmers)
  - Hard to navigate optimally by DBMS software
- Real value of ER is in using tables individually or in pairs
- Too complex for queries that span multiple tables with a large number of records

However, there is a paradox! In the fervor to make OLTP systems fast and efficient, the designers lost sight of the original, most important goal i.e. querying the databases to retrieve data/information. Thus this defeats the purpose as follows:

- Due to the sheer complexity of the ER graph, end users cannot understand or remember an ER model, and as a consequence cannot navigate an ER model. There is no graphical user interface (GUI) that takes a general ER model and makes it usable to end users, because this tends to be an NP-Complete problem.
- ER models are typically chaotic and "random", hence software cannot usefully query them. Cost-based optimizers that attempt to do this are infamous for making the wrong choices, and disastrous performance consequences.
- And finally, use of the ER modeling technique defeats the basic attraction of data warehousing, namely intuitive and high-performance retrieval of data.

ER vs. DM

ER	DM
Constituted to optimize OLTP performance.	Constituted to optimize DSS query performance.
Models the <u>micro</u> relationships among data elements.	Models the <u>macro</u> relationships among data elements with an overall <u>deterministic</u> strategy.
A wild variability of the structure of ER models.	All dimensions serve as equal entry points to the fact table.
Very vulnerable to changes in the user's querying habits, because such schemas are asymmetrical.	Changes in user querying habits can be catered by automatic SQL generators.

Table 13.1: ER vs. DM

1- ER models are constituted to (a) remove redundancy from the data model, (b) facilitate retrieval of individual records having certain critical identifiers, and (c) therefore, optimize On-line Transaction Processing (OLTP) performance.

2- ER modeling does not really model a business; rather, it models the micro relationships among data elements. ER modeling does not have "business rules," it has "data rules.“

3- The wild variability of the structure of ER models means that each data warehouse needs custom, handwritten and tuned SQL. It also means that each schema, once it is tuned, is very vulnerable to changes in the user's querying habits, because such schemas are asymmetrical.
===

1-In DM, a model of tables and relations is constituted with the purpose of optimizing decision support query performance in relational databases, relative to a measurement or set of measurements of the outcome(s) of the business process being modeled.

2-Even a big suite of dimensional models has an overall deterministic strategy for evaluating every possible query, even those crossing many fact tables.

3-All dimensions serve as equal entry points into the fact table. Changes in users' querying habits don't change the structure of the SQL or the standard ways of measuring and controlling performance.

How to simplify an ER data model?

- Two general methods:
  - De-Normalization
  - Dimensional Modeling (DM)

The shortcomings of ER modeling did not unnoticed. Since the beginning of the relational database revolution, many DB designers tried to deliver the design data to end users as rather look-alike "simpler designs with a "dimensional" i.e. ease of understanding and performance as the highest goals. There are actually two ways of “simplifying” the ER model i.e. (i) De-normalization and (ii) Dimensional Modeling.

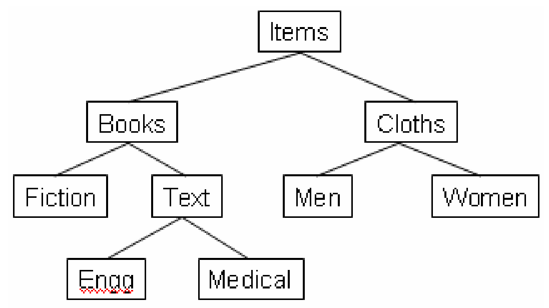
What is DM?…

- A simpler logical model optimized for decision support.
- Inherently dimensional in nature, with a single central fact table and a set of smaller dimensional tables.
- Multi-part key for the fact table
- Dimensional tables with a single-part PK.
- Keys are usually system generated.
-
- Results in a star like structure, called star schema or star join.
  - All relationships mandatory M-1.
  - Single path between any two levels.
- Supports ROLAP operations.

DM is a logical design technique that seeks to present the data in a standard, instinctive structure that supports high-performance and ease of understanding. It is inherently dimensional in nature, and it does adhere to the relational model, but with some important restrictions. Such as, every dimensional model is composed of one “central” table with a multipart key, called the fact table, and a set of smaller tables called dimension tables. Each dimension table has a single-part primary key that corresponds exactly to one of the components of the multipart key in the fact table. This results in a characteristic "star-like" structure or star schema.

DM is a logical design technique that seeks to present the data in a standard, instinctive structure that supports high-performance and ease of understanding. It is inherently dimensional in nature, and it does adhere to the relational model, but with some important restrictions. Such as, every dimensional model is composed of one “central” table with a multipart key, called the fact table, and a set of smaller tables called dimension tables. Each dimension table has a single-part primary key that corresponds exactly to one of the components of the multipart key in the fact table. This results in a characteristic "star-like" structure or star schema.

Dimensions have Hierarchies



Analysts tend to look at the data through dimension at a particular “level” in the hierarchy

Figure-13.2: Dimensions have Hierarchies

The foundation for design in this environment is through use of dimensional modeling techniques which focus on the concepts of “facts” and “dimensions” for organizing data.

Facts are the quantities or numerical measures (e.g., sales $) that we can count and the most useful being those that are additive. The most useful facts in a fact table are numeric and additive. Additive nature of facts is important, because data warehouse applications almost never retrieve a single record form the fact table; instead, they fetch back hundreds, thousands, or even millions of these records at a time, and the only useful thing to do with so many records is to add them up. Example, **what is the average salary of customers who’s age > 35 and experience more than 5 years?**

Dimensions are the descriptive textual information and the source of interesting constraints on how we filter/report on the quantities (e.g., by geography, product, date, etc.). For the DM shown, we constrain on the clothing department via the Dept attribute in the Product table. It should be obvious that the power of the database shown is proportional to the quality and depth of the dimension tables.

The two Schemas

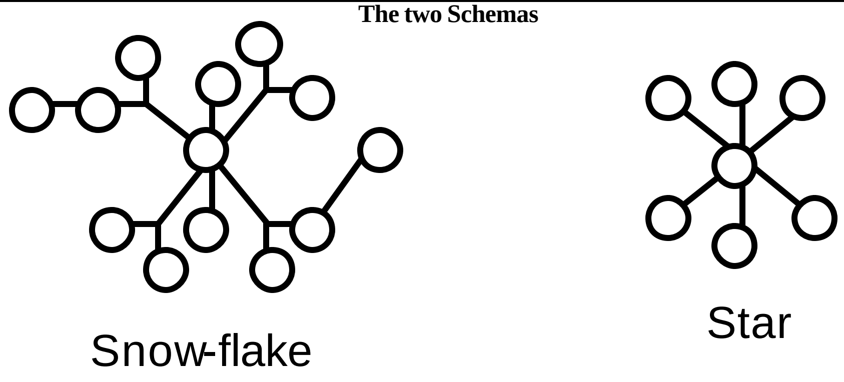


Figure-13.3: The two schemas

Fig-13.3 shows the snow-flake schema i.e. with multiple hierarchies that is typical of an OLTP or MIS system. The other is a simplified star schema with no hierarchies and a central node. Such schemas are typical of Data Warehouses.

Snowflake Schema: Sometimes a pure star schema might suffer performance problems. This can occur when a de-normalized dimension table becomes very large and penalizes the star join operation. Conversely, sometimes a small outer-level dimension table does not incur a significant join cost because it can be permanently stored in a memory buffer. Furthermore, because a star structure exists at the center of a snowflake, an efficient star join can be used to satisfy part of a query. Finally, some queries will not access data from outer-level dimension tables. These queries effectively execute against a star schema that contains smaller dimension tables. Therefore, under some circumstances, a snowflake schema is more efficient than a star schema.

Star Schema: A star schema is generally considered to be the most efficient design for two reasons. First, a design with de-normalized tables encounters fewer join operations. Second, most optimizers are smart enough to recognize a star schema and generate access plans that use efficient "star join" operations. It has been established that a "standard template" data warehouse query directly maps to a star schema.

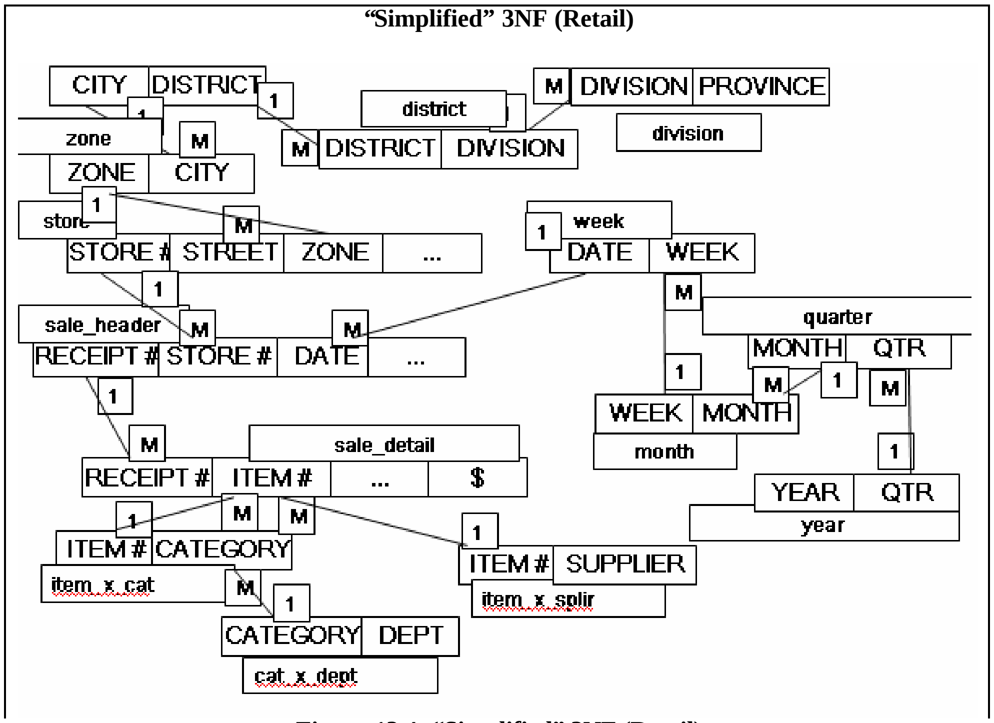


Figure-13.4: "Simplified" 3NF (Retail)

In Fig-13.4 a (simplified) retail data model is shown in the third normal form representation keeps each level of a dimensional hierarchy in a separate table (e.g., store, zone, region or item, category, department). The sale header and detail information is also maintained in two separate tables.

Vastly Simplified Star Schema

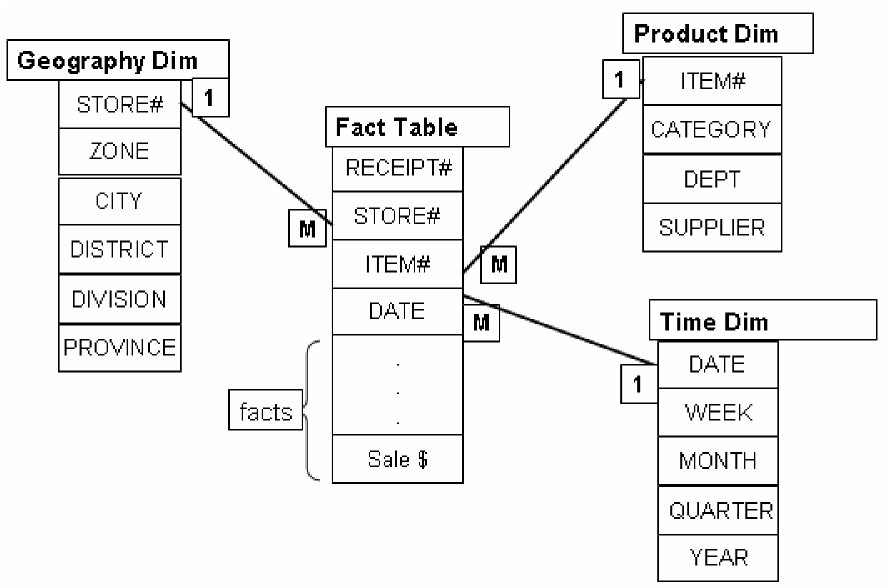


Figure-13.5: Vastly Simplified Star Schema

The goal of a star schema design is to simplify the physical data model so that RDBMS optimizers can exploit advanced indexing and join techniques in a straightforward manner, as shown in Fig-13.5. Some RDBMS products rely on star schemas for performance more than others (e.g., Red Brick versus Teradata).

The ultimate goal of a star schema design is to put into place a physical data model capable of very high performance to support iterative analysis adhering to an OLAP model of data delivery. Moreover, SQL generation is vastly simplified for front-end tools when the data is highly structured in this way.

In some cases, facts will also be summarized along common dimensions of analysis for additional performance.

The Benefit of Simplicity

Beauty lies in close correspondence with the business, evident even to business users.

The ultimate goal of a star schema design is to put into place a physical data model capable of very high performance to support iterative analysis adhering to an OLAP model of data delivery. Moreover, SQL generation is vastly simplified for front-end tools when the data is highly structured in this way.

In some cases, facts will also be summarized along common dimensions of analysis for additional performance.

Features of Star Schema

Dimensional hierarchies are collapsed into a single table for each dimension. Loss of information?

A single fact table created with a single header from the detail records, resulting in:

- A vastly simplified physical data model!
- Fewer tables (thousands of tables in some ERP systems).
- Fewer joins resulting in high performance.
- Some requirement of additional space.

By "flattening" the information for each dimension into a single table and combining header/detail records, the physical data model is vastly simplified. The simplified data model of a star schema allows for straightforward SQL generation and makes it easier for RDBMS optimizers detect opportunities for "star joins" as a means of efficient query execution.

Notice that star schema design is merely a specific methodology for deploying the pre-join de-normalization that we discussed earlier.

Quantifying space requirement

Quantifying use of additional space using star schema

There are about 10 million mobile phone users in Pakistan.
Say the top company has half of them = 500,000

Number of days in 1 year = 365
Number of calls recorded each day = 250,000 (assumed)
Maximum number of records in fact table = 91 billion rows
Assuming a relatively small header size = 128 bytes
Fact table storage used = 11 Tera bytes
Average length of city name = 8 characters $\approx$ 8 bytes
Total number of cities with telephone access = 170 (1 byte)
Space used for city name in fact table using Star = 8 x 0.091 = 0.728 TB
Space used for city code using snow-flake = 1x 0.091= 0.091 TB
Additional space used $\approx$ 0.637 Tera byte i.e. about 5.8%

By virtue of flattening the dimensions, instead of storing the city code, now in the "flattened" table the name of the city will be stored. There is a 1: 8 ratio between the two-representations. But out of a header size of 128, there has been an addition of 7 more bytes i.e. an increase in storage space of about 5%. This is not much, if there are frequent queries for which the join has not been eliminated.

Lecture-14
Process of Dimensional Modeling

The Process of Dimensional Modeling

Four Step Method from ER to DM

1. Choose the Business Process
2. Choose the Grain
3. Choose the Facts
4. Choose the Dimensions

A typical ER diagram covers the entire spectrum of the business, actually covers every possible business process. However, in reality those multitudes of process do not co-exist in time and space (tables). As a consequence, an ER diagram is overly complex, and is a demerit to itself. This is precisely the point that differentiates a DM from an ER diagram, as a single ER diagram can be divided into multiple DM diagrams. Thus a step-wise approach is followed to separate the DMs from an ER diagram, and this consists of four steps.

Step-1: Separate the ER diagram into its discrete business processes and to model each business process separately.

Step-2: Grain of a fact table = the meaning of one fact table row. Determines the maximum level of detail of the warehouse.

Step-3: Select those many-to-many relationships in the ER model containing numeric and additive non-key items and designate them as fact tables. Actually all business events to be analyzed are gathered into fact tables.

Step-4: De-normalize all of the remaining tables into flat tables with single-part keys that connect directly to the fact tables. These tables become the dimension tables. They are like reference tables that define how to analyze the fact information. They are typically small and relatively static.

Let's discuss each of the steps in detail, one by one.

Step-1: Choose the Business Process

- A business process is a major operational process in an organization.
- Typically supported by a legacy system (database) or an OLTP.
  - Examples: Orders, Invoices, Inventory etc.
- Business Processes are often termed as Data Marts and that is why many people criticize DM as being data mart oriented.

The first step in DM is the business process selection. What do we mean by a process? A process is a natural business activity in the organization supported by a legacy source data-collection system (database or OLTP). Example business processes include purchasing, orders, shipments, invoicing, inventory, and general ledger etc.

Many people consider business processes as Data marts i.e. in their view organizational or departmental function is referred as the business process. That is why such people criticize DM as being a data mart oriented approach. However, in Kimball's view, it is a wrong approach, the two must not be confused e.g. a single model is built to handle orders data rather than building separate models for marketing and sales departments, which both access the orders data. By focusing on business processes, rather than departments, consistent information can be delivered economically throughout the organization. Building departmental data models may result in data duplication, data inconsistencies, and data management issues. What is a possible solution? Yes, publishing data once can not only reduce the consistency problems, but can also reduce ETL development , data management and disk storage issues as well.

Step-1: Separating the Process

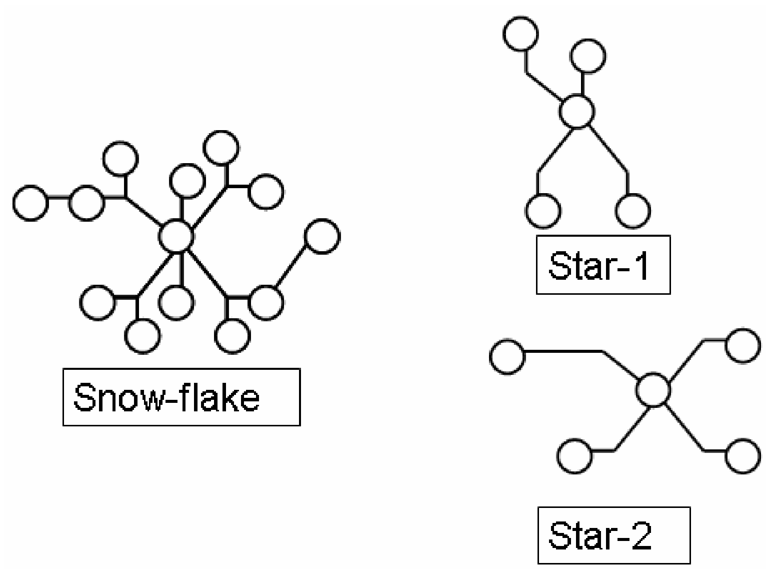


Figure-14.1: Step-1: Separating the Process

Fig-14.1 shows an interesting concept i.e. separating business processes to be modeled from a complex set of processes. This translates to splitting a snow-flake schema into multiple star schemas. Note that as the processes move into the star schema all the hierarchies collapse.

Step-2: Choosing the Grain

- Grain is the fundamental, atomic level ofdata to be represented.
- Grain is also termed as the unit of analyses.
- Example grain statements
- Typical grains
  - Individual Transactions
  - Daily aggregates (snapshots)
  - Monthly aggregates
- Relationship between grain and expressiveness.
- Grain vs. hardware trade-off.

Grain is the lowest level of detail or the atomic level of data stored in the warehouse. The lowest level of data in the warehouse may not be the lowest level of data recorded in the business system. It is also termed as the unit of analysis e.g. unit of weight is Kg etc.

Example grain statements: (*one fact row represents a…*)

- Entry from a cash register receipt
- Boarding pass to get on a flight
- Daily snapshot of inventory level for a product in a warehouse
- Sensor reading per minute for a sensor
- Student enrolled in a course

Finer-grained fact tables:

- are more expressive
- have more rows

Trade-off between performance and expressiveness

- Rule of thumb: Err in favor of expressiveness
- Pre-computed aggregates can solve performance problems

In the absence of aggregates, there is a potential to waste millions of dollars on hardware upgrades to solve performance problems that could have been otherwise addressed by aggregates.

Step-2: Choosing the Grain

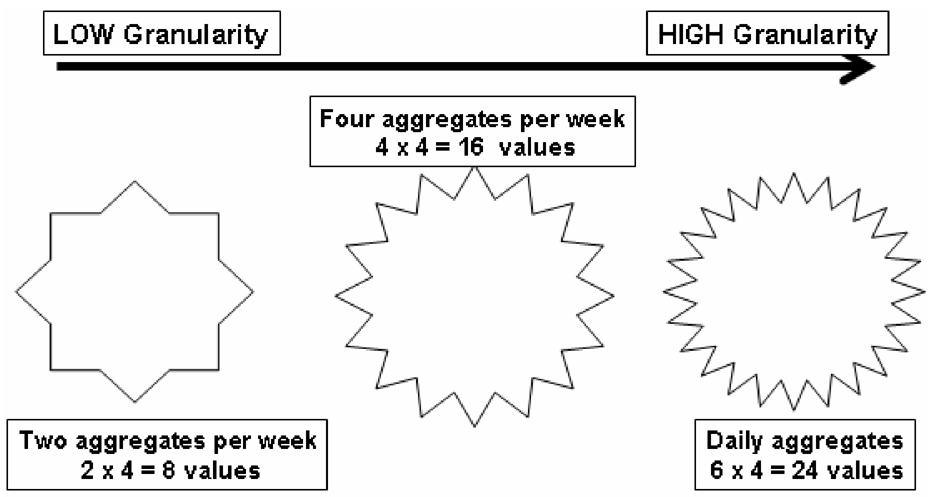


Figure-14.2: Step-2: Choosing the Grain

Note that you may come across definitions of grain as given in the notes and discussed in the lectures, but you may also come across definitions that are different from those discussed. This depends on the interpretation of the writer. We will follow the definition as per Fig-14.2.

The case FOR data aggregation

- Works well for repetitive queries.
- Follows the known thought process.
- Justifiable if used for max number of queries.
- Provides a "big picture" or macroscopic view.
- Application dependent, usually inflexible to business changes (remember lack of absoluteness of conventions).

There are both positives and negatives to data aggregation. These are a list of the reasons for the utilization of summary or aggregate data. As you can see, they all really fall under the area of "performance".

The negative side is that summary data does not allow a total solution with the flexibility and capabilities that some businesses truly require as compared to other businesses.

The case AGAINST data aggregation

- Aggregation is irreversible.
  - Can create monthly sales data from weekly sales data, but the reverse is not possible.

- Aggregation limits the questions that can be answered.
  - What, when, why, where, what-else, what-next

- Aggregation can hide crucial facts.
  - The average of 100 & 100 is same as 150 & 50

Aggregation is one-way i.e. you can create aggregates, but can not dissolve aggregates to get the original data from which the aggregates were created. For example 3+2+1 = 6 at the same time 2+4 also equals 6, so does 5+1 and if we consider reals, then infinetly many ways of adding numbers to get the same result.

If you think about the “5 W’s of journalism”, these are the “6 W’s of data analysis”. Again it highlights the types of questions that end users want to ask and can not be answered by summary data.

By definition, a summarization will consider at least one of these points irrelevant. For example, a summary across the company takes out the dimension of “WHERE” and a summary by quarters takes out the element of “WHEN”. The point to be noted is that although summary data has a purpose, yet one can take any summary and ask a question that the system can not answer.

Aggregation hides crucial facts

Just Looking at the averages i.e. aggregates

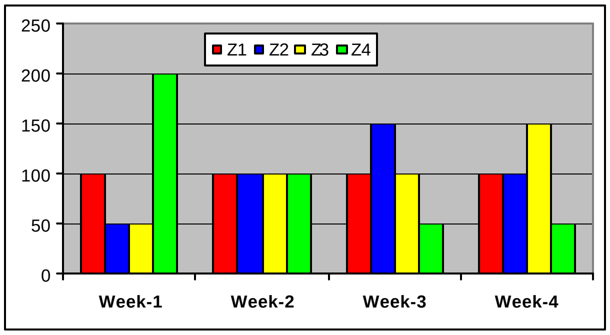
	Week-1	Week-2	Week-3	Week-4	Average
Zone-1					100
Zone-2					100
Zone-3					100
Zone-4					100
Average	100	100	100	100	

Table-14.1: Aggregation hides crucial facts

Consider the sales data of an item sold in a chain store in four zones, such that the sales data is aggregated across the weeks also. For this simple example, for the sake of conserving space the average sales across each zone and for each week is stored. Therefore, instead of storing 16 values only 8 values are stored i.e. a saving of 50% space.

Assume that a promotional scheme or advertisement campaign was run, and then the sales data was recorded to analyze the effectiveness of the campaign. If we look at the averages (as shown in the table) there is no change in sales i.e. neither across time nor across the geography dimension. On the face of it, it was an in effective campaign. Now lets raise the curtain and look at the detailed sales records. The numbers are NOT constant! Drawing the graphs of the sales records, shows a very different picture.

Aggregation hides crucial facts



Z1: Sale is constant (need to work on it)
Z2: Sale went up, then fell (need of concern)
Z3: Sale is on the rise, why?
Z4: Sale dropped sharply, need to look deeply.
W2: Static sale

Figure-14.3: Aggregation hides crucial facts

Z1: Sale is constant through out the month (need to work on it)

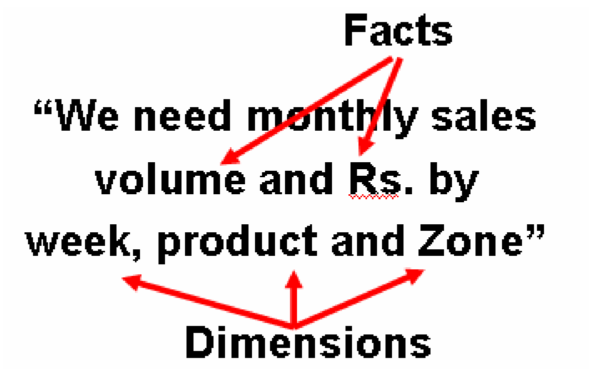
Z2: Sale went up, then fell (need of concern) i.e. the campaign was effective, but after week it fizzled down.

Z3: Sale is on the rise, why?

Z4: Sale dropped sharply, need to look deeply. It seems that the campaign had a negative effect on the sales?

W2: Static sale across all zones, very unique indeed.

Step 3: Choose Facts



Facts
"We need monthly sales volume and Rs. by week, product and Zone"
Dimensions

Numeric facts are identified by answering the question "what are we measuring?" Many- to-many relationships in the ER model containing numeric and additive non-key items are selected and designated as fact tables. In the example numeric additive figures *volume* (quantity ordered) and *Rs*. (Rupees cost amount) are the facts because the numeric values of the two are of keen interest for the business user.

Step 3: Choose Facts

- Choose the facts that will populate each fact table record.
  - Remember that best Facts are Numeric, Continuously Valued and Additive.
  - Example: Quantity Sold, Amount etc.

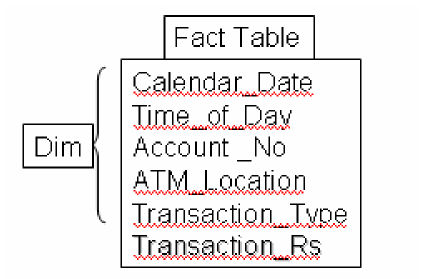
It should be remembered that facts are numeric, continuous, additive and non-key items that will populate the fact table. Example facts for a point of sales terminal (POS) are sales quantity, per unit sales price, and the sales amount in rupees. All the candidate facts in a design must be true to the grain described in previous slides. Facts that clearly belong to a different grain must be in a separate fact table.

Step 4: Choose Dimensions

- Choose the dimensions that apply to each fact in the fact table.
  - Typical dimensions: time, product, geography etc.
  - Identify the descriptive attributes that explain each dimension.
  - Determine hierarchies within each dimension.

Step-4: How to Identify a Dimension?

- The single valued attributes during recording of a transaction are dimensions.



Time_of_day: Morning, Mid Morning, Lunch Break etc.
Transaction_Type: Withdrawal, Deposit, Check balance etc.

Table-14.2: Step-4: How to Identify a Dimension?

The dimension tables, usually represent textual attributes that are already known about things such as the product, the geography, or the time. If the database designer is very clear about the grain of the fact table, then choosing the appropriate dimensions for the fact table is usually easy. What is so special about it, seems to be pretty intuitive, but is not.

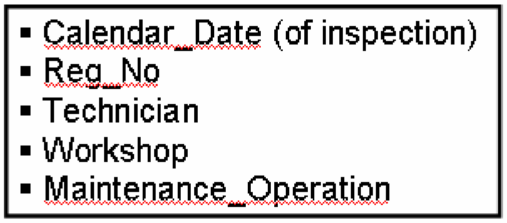
The success in selecting the right dimensions for a given fact table is dependent on correctly identifying any description that has a single value for an individual fact table record or transaction. Note that the fact table record considered could be a single transaction or weekly aggregate or monthly sums etc i.e. a grain is associated. Once this is correctly identified and settled, then as many dimensions can be added to the fact able as required. For the ATM customer transaction example, the following dimensions all have a single value during the recording of the transaction, as none of the above dimensions change during a single transaction:

- Calendar_Date
- Time_of_Day
- Account _No
- ATM_Location
- Transaction_Type (withdrawal, deposit, balance inquiry etc.)

Over here Time_of_Day refers to specific periods such as Morning, Mid Morning, Lunch Break, Office_Off etc. Note that during an atomic transaction, the value of Time_of_Day does not change (as a transaction takes less than a minute), hence it is a dimension. In the context of the ATM example, the only numeric attribute is the Transaction_Rs, so it is a fact. Observe that we use this convention in real life also, when people say we will visit you first time or second time of the day etc.

Step-4: Can Dimensions be Multi-valued?

- Are dimensions ALWYS single?
  - Not really
  - What are the problems? And how to handle them



- Calendar_Date (of inspection)
- Reg_No
- Technician
- Workshop
- Maintenance_Operation

- How many maintenance operations are possible?
  - Few
  - Maybe more for old cars

Figure-14.4: Step-4: Can Dimensions be Multi-valued?

After convincing ourselves that dimensions are really single valued, perhaps we should consider whether there are ever legitimate exceptions i.e. is it possible to have multi-valued dimension in a fact table? If this is conceivable, what problems might arise?

Consider the following example from vehicle maintenance system used at a vehicle service center. You are handed a data sheet for which the grain is the individual line item on the customer's bill. The data source could be your periodic car maintenance visits to the company workshop or individual replacement charges on a repair/change bill. These individual line items have a rich set of dimensions such as:

- Calendar_Date (of inspection)
- Reg_No (of vehicle)
- Technician_ID
- Workshop
- Maintenance_Operation

The numeric additive facts in this design (which are the core of every fact table in a dimensional design) would include Amount_Charged and perhaps others including Amount_Paid, depending upon if the vehicle was insured etc.

On the face of it, this seems to be a very straightforward design, with obvious single values for all the dimensions. But there is a surprise. In many situations, there may be multiple values for services performed, such as oil change, air filter change, spark plug change etc. What do you do if for a certain car there are three separate changes at the moment the service was performed? How about really old and ill-kept cars that might have upto 5+ such changes? How do you encode the Maintenance_Operation dimension if you wish to represent this information?

Step-4: Dimensions & Grain

- Several grains are possible as per business requirement.
    - For some aggregations certain descriptions do not remain atomic.
    - Example: Time_of_Day may change several times during daily aggregate, but not during a transaction
- Choose the dimensions that are applicable within the selected grain.

Strangely, there is a relationship between the grain and the dimensions. When building a fact table, the most important step is to declare the grain (aggregation level) of the fact table. The grain declares the exact meaning of an individual fact record. Consider the case of transactions for an ATM machine. The grain could be individual customer transaction, or number of transaction per week or the amount drawn per month.

- Calendar_Date
- Time_of_Day
- Account _No
- ATM_Location
- Transaction_Type (withdrawal, deposit, balance inquiry etc.)

Note that none of the above dimensions change during a single transaction. However, for weekly transactions probably only Account _No and ATM_Location can be treated as a dimension.

Note that higher the level of aggregation of the fact table, the fewer will be the number of dimensions you can attach to the fact records. The converse of this is surprising. The more granular the data, the more dimensions make sense. Hence the lowest-level data in any organization is the most dimensional.

Lecture-15
Issues of Dimensional Modeling

Step 3: Additive vs. Non-Additive facts

- Additive facts are easy to work with
  - Summing the fact value gives meaningful results
  - Additive facts:
    - Quantity sold
    - Total Rs. sales
  - Non-additive facts:
    - Averages (average sales price, unit price)
    - Percentages (% discount)
    - Ratios (gross margin)
    - Count of distinct products sold

Month	**Crates of Bottles Sold**
May	14
Jun.	20
Jul.	24
TOTAL	58

Month	**% discount**
May	10
Jun.	8
Jul.	6
TOTAL	24% ? Incorrect!

There can be two types of facts i.e. additive and non-additive. Additive facts are those facts which give the correct result by an addition operation. Examples of such facts could be number of items sold, sales amount etc. Non-additive facts can also be added, but the addition gives incorrect results. Some examples of non-additive facts are average, discount, ratios etc. Consider three instances of 5, with the sum being 15 and average being 5. Now consider two numbers i.e. 5 and 10, the sum being 15, but the average being 7.5. Now if the average of 5 and 7.5 is taken this comes to be 6.25, but if the average of the actual numbers is taken, the sum comes to be 30 and the average being 6. Hence averages, if added gives wrong results. Now facts could be averages, such as average sales per week etc, thus they are perfectly legitimate facts.

Step-3: Classification of Aggregation Functions

- How hard to compute aggregate from sub-aggregates?
  - Three classes of aggregates:
    - Distributive
      - Compute aggregate directly from sub-aggregates
      - Examples: MIN, MAX ,COUNT, SUM
    - Algebraic
      - Compute aggregate from constant-sized summary of subgroup
      - Examples: STDDEV, AVERAGE
      - For AVERAGE, summary data for each group is SUM, COUNT

- Holistic
    - Require unbounded amount of information about each subgroup
    - Examples: MEDIAN, COUNT DISTINCT
    - Usually impractical for a data warehouses!

We see that calculating aggregates from aggregates is desirable, but is not possible for non-additive facts. So we deal with three types of aggregates i.e. distributive that are additive in nature, and then algebraic which are non-additive in nature. Therefore, such aggregates have to be computed from summary of subgroups to avoid the problem of incorrect results. The of course are the holistic aggregates that give a complete picture of the data, such as median, or distinct values. However, such aggregates are not desirable for a data warehouse environment, as it requires a complete scanning, which is highly undesirable as it consumes lot of time.

Step-3: Not recording Facts

- Transactional fact tables don't have records for events that don't occur
    - Example: No records(rows) for products that were not sold.
- This has both advantage and disadvantage.
    - Advantage:
        - Benefit of sparsity of data
        - Significantly less data to store for "rare" events
    - Disadvantage: Lack of information
    - Example: What products on promotion were not sold?

Fact tables usually don't records events that don't happen, such as items that were not sold. The advantage of this approach is getting around the problem of sparsity. Recall that when we discussed MOLAP, we discussed the sales of different items not occurring in different geographies and in different time frames, resulting in sparse cubes. If however this data is not recorded, then significantly less data will be required to be stored. But what if, from the point of view of decision making, such data has to be retrieved, how to retrieve data corresponding to those items? To find such items, additional queries will be required to check the current item balance with the item balance when the items where (say) brought into the store. So the biggest disadvantage of this approach is key data is not recorded.

Step-3: A Fact-less Fact Table

- "Fact -less" fact table
    - A fact table without numeric fact columns
    - Captures relationships between dimensions
    - Use a dummy fact column that always has value 1

The problem of not recording non-events is solved by using fact-less fact tables, as not recording such information resulted in loss of data. Such a fact-less fact table is one which does not have numeric values stored in the corresponding column, as such tables are used to capture the relationships between dimensions. Fact less fact table captures the many-to-many relationships between dimensions, but contains no numeric or textual facts. To achieve this dummy value of 1 is used in the corresponding column.

Step-3: Example: Fact-less Fact Tables

Examples:

- Department/Student mapping fact table
    - What is the major for each student?
    - Which students did not enroll in ANY course
- Promotion coverage fact table
    - Which products were on promotion in which stores for which days?
    - Kind of like a periodic snapshot fact

Some of the examples of fact-less fact tables. Consider the case of a department/student mapping fact table. The data is recorded for each student who registers for a course, but there may be students that do not register in any course. If data is useful from the point of view of identifying those students which are skipping a semester. There is no direct or simple way to identify such students, the solution is a fact-less fact table. Similarly which items on promotion are not selling, as the sales records are for only those items that are sold.

Step-4: Handling Multi-valued Dimensions?

- One of the following approaches is adopted:
    - Drop the dimension.
    - Use a primary value as a single value.
    - Add multiple values in the dimension table.
    - Use “Helper” tables.

For handling the exceptions in dimensions, designers adopt one of the following approaches:

- Drop the Maintenance_Operation dimension as it is multi-valued.
- Choose one value (as the "primary" maintenance) and omit the other values.
- Extend the dimension list and add a fixed number of maintenance dimensions.
- Put a helper table in between this fact table and the Maintenance dimension table.

Instead of ignoring the problem and dropping the dimension altogether, let’s tackle the problem.

Usually the designers go for the second alternative, as a consequence this will show up as the primary, or main maintenance operation. Such as 20,000 Km maintenance or 40,000 Km maintenance. It is known what would constitute for each mileage based maintenance, and these maintenance are also mutually exclusive i.e. single valued. In many cases, you may actually come across this practice being observed in the OLTP systems. The obvious advantage is that the modeling problem is resolved, but the disadvantage is that the usefulness of the data becomes questionable. Why? Because with the passage of time or with new models of vehicles coming or because of company policy, what constitutes service at 20,000 Km may actually change. Will need meta data to resolve this issue.

The third alternative of creating a fixed number of additional columns in the dimension table is a quick and dirty approach and should be avoided. There is likely to be car that may require more changes then reflected in the table, or the company policy changes and more items fall under the maintenance, and a long list will result in many null entries for a typical car, especially new ones. Furthermore, it is not easy to query the multiple separate maintenance dimensions and will result in slow queries. Therefore, multiple dimensions style of design should be avoided.

The last alternative is usually adopted, and a "helper" table is placed between the Maintenance dimension and the fact table, although this adulterates or dilutes the star schema. More details are beyond the scope of this course.

Step-4: OLTP & Slowly Changing Dimensions

OLTP systems not good at tracking the past. History never changes.

OLTP systems are not “static” always evolving, data changing by overwriting.

Inability of OLTP systems to track history, purged after 90 to 180 days.

Actually don’t want to keep historical data for OLTP system.

One major difference between an OLTP system and a data warehouse is the ability and the responsibility to accurately describe the past. OLTP systems are usually very poor at correctly representing a business as of a month or a year ago for several reasons as discussed before. A good OLTP system is always evolving. Orders are being filled and, thus, the order backlog is constantly changing. Descriptions of products, suppliers, and customers are constantly being updated, usually by overwriting. The large volume of data in an OLTP system is typically purged every 90 t o 180 days. For these reasons, it is difficult for an OLTP system to correctly represent the past. In an OLTP system, do you really want to keep old order statuses, product descriptions, supplier descriptions, and customer descriptions over a multiyear period?

Step-4: DWH Dilemma: Slowly Changing Dimensions

The responsibility of the DWH to track the changes.

Example: Slight change in description, but the product ID (SKU) is not changed.

Dilemma: Want to track both old and new descriptions, what do they use for the key? And where do they put the two values of the changed ingredient attribute?

The data warehouse must accept the responsibility of accurately describing the past. By doing so, the data warehouse simplifies the responsibilities of the OLTP system. Not only does the data warehouse relieve the OLTP system of almost all forms of reporting, but the data warehouse contains special structures that have several ways of tracking historical data.

A dimensional data warehouse database consists of a large central fact table with a multipart key. This fact table is surrounded by a single layer of smaller dimension tables, each containing a single primary key. In a dimensional database, these issues of describing the past mostly involve slowly changing dimensions. A typical slowly changing dimension is a product dimension in which the detailed description of a given product is occasionally adjusted. For example, a minor ingredient change or a minor packaging change may be so small that production does not assign the product a new SKU number (which the data warehouse has been using as the primary key in the product dimension), but nevertheless gives the data warehouse team a revised description of t he product. The data warehouse team faces a dilemma when this happens. If they want the data warehouse to track both the old and new descriptions of the product, what do they use for the key? And where do they put the two values of the changed ingredient

Step-4: Explanation of Slowly Changing Dimensions…

- Compared to fact tables, contents of dimension tables are relatively stable.
  - New sales transactions occur constantly.
  - New products are introduced rarely.
  - New stores are opened very rarely.

- The assumption does not hold in some cases
  - Certain dimensions evolve with time
  - e.g. description and formulation of products change with time
  - Customers get married and divorced, have children, change addresses etc.
  - Land changes ownership etc.
  - Changing names of sales regions.

For example, a minor ingredient change or a minor packaging change may be so small that production does not assign the product a new SKU number (which the data warehouse has been using as the primary key in the product dimension), but nevertheless gives the data warehouse team a revised description of t he product. The data warehouse team faces a dilemma when this happens. If they want the data warehouse to track both the old and new descriptions of the product, what do they use for the key? And where do they put the two values of the changed ingredient attribute?

Other common slowly changing dimensions are the district and region names for a sales force. Every company that has a sales force reassigns these names every year or two. This is such a common problem that this example is something of a joke in data ware housing classes. When the teacher asks, "How many of your companies have changed the organization of your sales force recently?" everyone raises their hands.

Step-4: Explanation of Slowly Changing Dimensions…

Although these dimensions change but the change is not rapid.

Therefore called “Slowly” Changing Dimensions

There can be many examples. For a young customer who is single, then after a while the customer gets married. After sometime there are children, in unfortunate cases the marriage breaks so the customer is separated or the husband dies and the customer becomes a widow. This just does not typically happen overnight but takes a while. Another example is inheritance, consider the example of land. Over a period of time the land changes hands, is split because of inheritance or its size increases by buying. Again things don't happen overnight, but take a while, hence slowly changing dimensions.

Step-4: Handling Slowly Changing Dimensions

- Option-1: Overwrite History
  - Example: Code for a city, product entered incorrectly
  - Just overwrite the record changing the values of modified attributes.
  - No keys are affected.
  - No changes needed elsewhere in the DM.
  - Cannot track history and hence not a good option in DSS.

The first technique is the simplest and fastest. But it doesn't maintain past history! Nevertheless, overwriting is frequently used when the data warehouse team legitimately decides that the old value of the changed dimension attribute is not interesting . For example, if you find incorrect values in the city and state attributes in a customer record, then overwriting would almost certainly be used. After the overwrite, certain old reports that depended on the city or state values would not return exactly the same values. Most of us would argue that this is the correct outcome.

Step-4: Handling Slowly Changing Dimensions

- Option-2: Preserve History
  - Example: The packaging of a part change from glued box to stapled box, but the code assigned (SKU) is not changed.
  - Create an additional dimension record at the time of change with new attribute values.
  - Segments history accurately between old and new description
  - Requires adding two to three version numbers to the end of key. SKU#+1, SKU#+2 etc.

Suppose you work in a manufacturing company and one of your main data warehouse schemas is the company's shipments. The product dimension is one of the most important dimensions in this dimensional schema. A typical product dimension would have several hundred detailed records, each representing a unique product capable of being shipped. A good product dimension table would have at least 50 attributes describing the products, including hierarchical attributes such as brand and category, as well as nonhierarchical attributes such as color and package type. An important attribute provided by manufacturing operations is the SKU number assigned to the product. You should start by using the SKU number as the key to the product dimension table. Suppose that manufacturing operations makes a slight change in packaging of SKU #38, and the packaging description changes from "glued box" to "pasted box." Along with this change, manufacturing operations decides not to change the SKU number of the product, or the bar code (UPC) that is printed on the box. If the data warehouse team decides to track this change, the best way to do this is to issue another product record, as if the pasted box version were a brand new product. The only difference between the two product records is the packaging description. Even the SKU numbers are the same. The only way you can issue another record is if you generalize the key to the product dimension table to be something more than the SKU number. A simple technique is to use the SKU number plus two or three version digits. Thus the first instance of the product key for a given SKU might be SKU# + 01. When, and if, another version is needed, it becomes SKU# + 02, and so on. Notice that you should probably also park t he SKU number in a separate dimension attribute (field) because you never want an application to be parsing the key to extract the underlying SKU number. Note the separate SKU attribute in the Product dimension in Figure 1.

This technique for tracking slowly changing dimensions is very powerful because new dimension records automatically partition history in the fact table. The old version of the dimension record points to all history in the fact table prior to the change. The new version of the dimension record points to all history after the change. There is no need for a timestamp in the product table to record the change. In fact, a timestamp in the dimension record may be meaningless because the event of interest ist he actual use of the new product type in a shipment. This is best recorded by a fact table record with the correct new product key.

Another advantage of this technique is that you can gracefully track as many changes to a dimensional item as you wish. Each change generates a new dimension record, and each record partitions history perfectly. The main drawbacks of the technique are the requirement to generalize the dimension key, and the growth of the dimension table itself.

Step-4: Handling Slowly Changing Dimensions

- Option-3: Create current valued field
    - Example: The name and organization of the sales regions change over time, and want to know how sales would have looked with old regions.
    - Add a new field called current_region rename old to previous_region.
    - Sales record keys are not changed.
    - Only TWO most recent changes can be tracked.

Creating a Current Value Field

You use the third technique when you want to track a change in a dimension value, but it is legitimate to use the old value both before and after the change. This situation occurs most often in the infamous sales force realignments, where although you have changed the names of your sales regions, you still have a need to state today's sales in terms of yesterday's region names, just to "see how they would have done" using the old organization. You can attack this requirement, not by creating a new dimension record as in the second technique, but by creating a new "current value" field. Suppose in a sales team dimension table, where the records represent sales teams, you have a field called "region." When you decide to rearrange the sales force and assign each team to newly named regions, you create a new field in the sales dimension table called "current_region." You should probably rename the old field "previous_region." No alterations are made to the sales dimension record keys or to the number of sales team records. These two fields now allow an application to group all sales fact records by either the old sales assignments (previous region) or the new sales assignments (current region). This schema allows only the most recent sales force change to be tracked, but it offers the immense flexibility of being able to state all of the history by either of the two sales force assignment schemas. It is conceivable, although somewhat awkward, to generalize this approach to the two most recent changes. If many of these sales force realignments take place and it is desired to track them all, then the second technique should probably be used.

Step-4: Pros and Cons of Handling

- Option-1: Overwrite existing value
    - + Simple to implement
    - + No tracking of history
- Option-2: Add a new dimension row
    - + Accurate historical reporting
    - + Pre-computed aggregates unaffected
    - + Dimension table grows over time
- Option-3: Add a new field
    - + Accurate historical reporting to last TWO changes
    - + Record keys are unaffected
    - + Dimension table size increases

There are number of ways of handling slowly changing dimensions. Some of the methods are simple, but not desirable; but all have their own pros and cons. The simplest possible "solution" is to overwrite history. If the customer was earlier single, and gets married, just change his/here status from single to married. Very simple to implement, but not desirable, as a DWH is about recording historical data, and by virtue of overwriting, the historical data is destroyed. Another option is to add a row when the dimension changes, the obvious benefit is that history is not lost, but over the period of time the dimension table will grow as new rows are added corresponding to the changes in the dimensions. The third, rather desirable approach is to add an additional column, that does increase the table size, but the increase is not non-deterministic. The column records the last two changes, if the dimension changes more than twice, then historical data is lost.

Step-4: Junk Dimension

- Sometimes certain attributes don't fit nicely into any dimension
    - Payment method (Cash vs. Credit Card vs. Check)
    - Bagging type (Paper vs. Plastic vs. None)
- Create one or more "mix" dimensions
    - Group together leftover attributes as a dimension even if not related
    - Reduces number of dimension tables, width of fact table
    - Works best if leftover attributes are
        - Few in number
        - Low in cardinality
        - Correlated
- Other options
    - Each leftover attribute becomes a dimension
    - Eliminate leftover attributes that are not useful

A junk dimension is a collection of random transactional codes, flags and/or text attributes that are unrelated to any particular dimension. The junk dimension is simply a structure that provides a convenient place to store the junk attributes. A good example would be a trade fact in a company that brokers equity trades.

The need for junk dimensions arises when we are considering single level hierarchies. The only problem with the single level hierarchies is that you may have a lot of them in any given dimensional model. Ideally, the concatenated primary key of a fact table should consist of fewer than 10 foreign keys. Sometimes, if all of the yes/no flags are represented as single level hierarchy dimensions, you may end up with 30 or more. Obviously, this is an overly complex design.

A technique that allows reduction of the number of foreign keys in a fact table is the creation of "junk" dimensions. These are just "made up" dimensions where you can put several of these single level hierarchies. This cuts down the number of foreign keys in the fact table dramatically.

As to the number of flags before creating a junk dimension, if there are more than 15 dimensions, where five or more are single level hierarchies, I start seriously thinking about combining them into one or more junk dimensions. One should not indiscriminately combine 20 or 30 or 80 single level hierarchies.

Lecture-16
Extract Transform Load (ETL)

In a DWH project, 50% of the work is related to ETL. ETL stands for Extract Transform Load. Some people call it ETML i.e. Extract Transform Move Load, as you have to move the data into the DWH. There is a significant part after data transformation that involves data cleansing i.e. there is a C also here. However, ETL is a pretty standard definition, and normally when you say ETL people know what you mean.

E: Extract from the operational system
T: Transform the data, which includes data cleansing
L: Loading the data into the DWH

Putting the pieces together

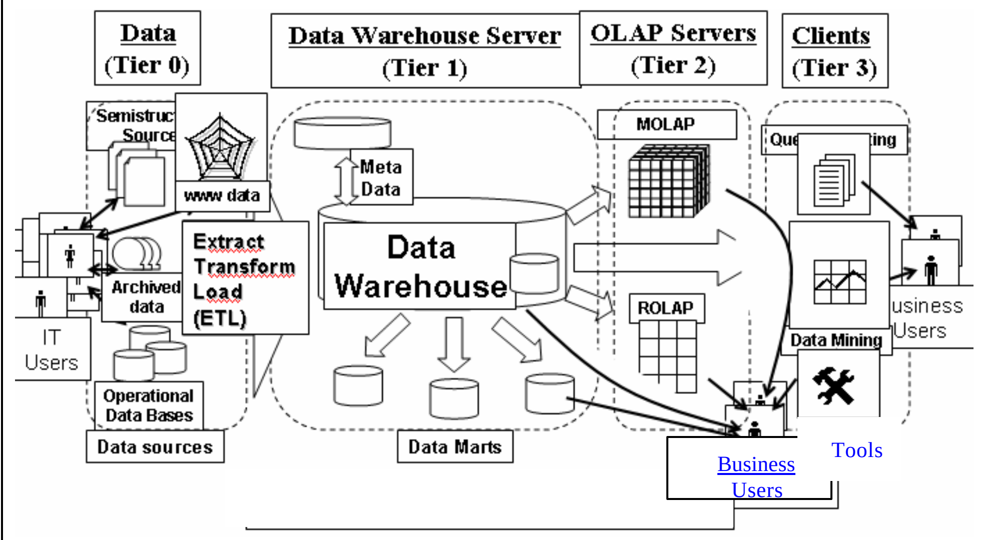


Figure-16.1: ETL: Putting the pieces together

Figure-16.1 shows a multi-tiered data warehousing architecture. We can see that at the lowest level (tier 0) lies the data sources like operational, archived and/or web etc. Data Warehouse Server lays in the next level (tier 1). We can see in the Figure 16.1 that ETL box lies at the boundaries of both the tiers 0 and 1. That means it serves as a bridgebetween data source systems and the actual data warehouse. Thus the source data is processed using some set of activities before bringing it into the data warehouse environment. These set of activities comprise the ETL process.

The ETL Cycle

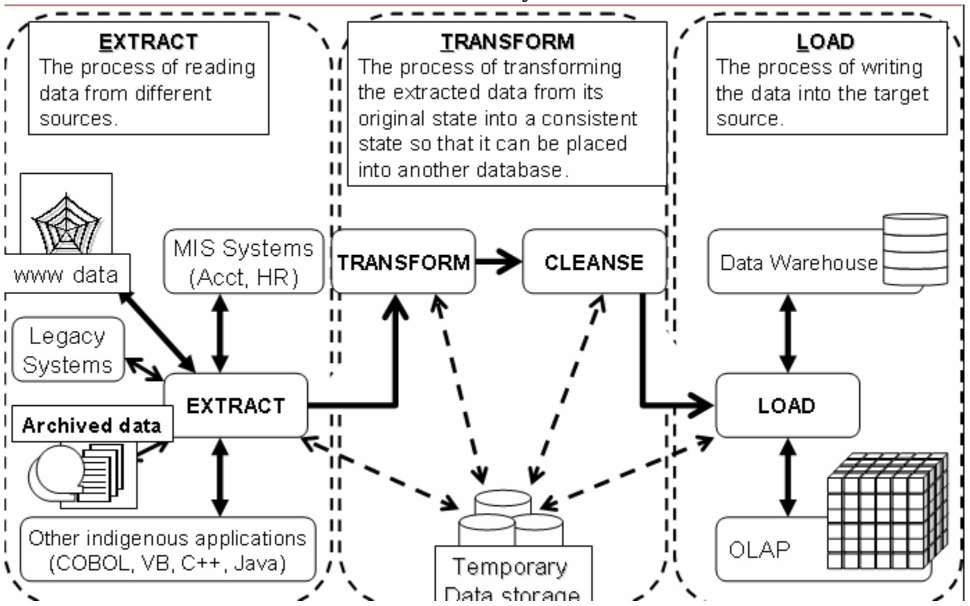

EXTRACT
The process of reading data from different sources.
TRANSFORM
The process of transforming the extracted data from its original state into a consistent state so that it can be placed into another database.
LOAD
The process of writing the data into the target source.
www data
MIS Systems (Acct, HR)
Legacy Systems
Archived data
Other indigenous applications (COBOL, VB, C++, Java)
EXTRACT
TRANSFORM
CLEANSE
Temporary Data storage
Data Warehouse
LOAD
OLAP

Figure-16.2: The ETL Cycle

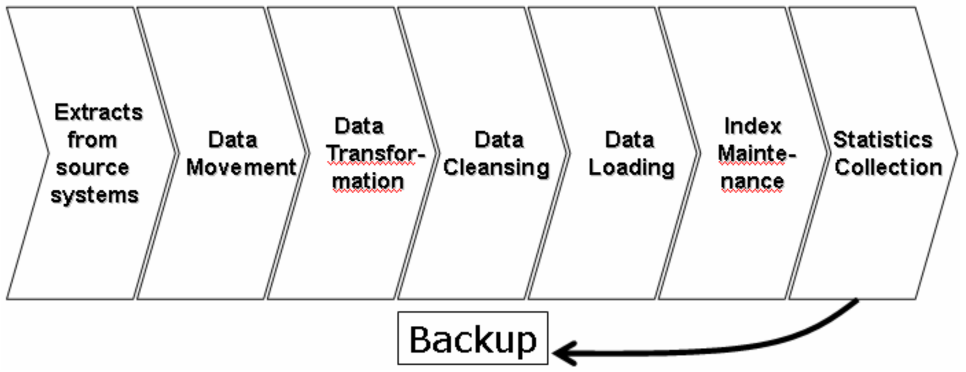
Now what are those activities in the ETL process shown as a box in the Figure 16.1? Let's have a look inside the box to find the sequence of activities that constitute the ETL process. As shown in Figure 16.2 ET L process consists of three major activities separated by dotted lines. The first step is the *Extract* which is the activity of reading data from multiple heterogeneous data sources. Next comes the *Transform* step which is the activity of transforming the input data from multiple heterogeneous environments into a single consistent state. Another important activity at this level is the *data cleansing* which is the activity of noise removal from input data before bringing it in the DWH environment. The final step is the *Load* which is an activity of loading cleansed data in to the target system.

ETL Processing

ETL is independent yet interrelated steps.

It is important to look at the big picture.

Data acquisition time may include…



Back-up is a major task, its a DWH not a cube

Figure-16.3: ETL Processing

When we look at ETL processing, it is important to look at the big picture as shown in Figure 16.3. It is not just the transformation; there is a whole lot of work you have to consider in the design and architecture of the ETL model. You have to design to extract from source system, how to get the data out of the operational databases. You have to move the data to the dedicated server where the transformations are being done or on the very least if the transformations will be done on the same systems, start moving the data into the warehouse environment. You have to look at the data movement and all the network implications of data movement. The transformations we will be talking about and the data loading strategies all have to be considered. It is not just good enough to talk about data loading, we have to do index maintenance, statistics collection, summary, data maintenance, data mart construction, data back ups, a whole process has to be followed each time you refresh or reload the data warehouse.

The architects really need to consider all of these. We will be focusing on all the major operations of ETL, so as to understand the big picture. It is certainly true that the volume of the data warehouse will make a difference on how you do the backups and the technology. Consider a cube which is not a data warehouse, it is a data mart. Typically a data mart is much smaller and probably pretty easier to backup just part of a file system backu p on wherever your cube server is. The data warehouse is usually much bigger. So for a data warehouse you usually do a full tape backup, and a tape-backup would ideally use a robotic tape library and all the associated setup. Typically you don't need robotic libraries for the cubes and data marts, but you do need them for data warehouse. But again you have to come back to economics, robotic tape libraries are very expensive, and so you may actually use manual tape mounting, because it is not worth paying for robots when you can have humans do it on a very different price range.

Overview of Data Extraction

First step of ETL, followed by many.

Source systems for extraction are typically OLTP systems.

A very complex task due to number of reasons:

- Very complex and poorly documented source system.
- Data has to be extracted not once, but number of times.

The process design is dependent on:

- Which extraction method to choose?
- How to make available extracted data for further processing?

Extraction is the operation of extracting data from a source system for further use in a data warehouse environment. This is the first step of the ETL process. After the extraction, this data can be transformed, cleansed and loaded into the data warehouse.

The source systems for a data warehouse are typically transaction processing applications. For example, one of the source systems for a sales analysis data warehouse might be an order entry system that records all of the current order activities.

Designing and creating the extraction process is often one of the most time-consuming tasks in the ETL process and, indeed, in the entire data warehousing process. The source systems might be very complex and poorly documented, and thus determining which data needs to be extracted can be difficult. The data has to be extracted normally not only once, but several times in a periodic manner to supply all changed data to the warehouse and keep it up-to-date. Moreover, the source system typically cannot be modified, nor can its performance or availability be adjusted, to accommodate the needs of the data warehouse extraction process. These are important considerations for extraction and ETL in general.

Designing this process means making decisions about the following two main aspects:

- Which extraction method to choose?

This influences the source system and the time needed for refreshing the warehouse.

- How to make available extracted data for further processing?

This influences the transportation method, and the need for cleaning and transforming the data.

Types of Data Extraction

- **Logical Extraction**
  - Full Extraction
  - Incremental Extraction

- **Physical Extraction**
  - Online Extraction
  - Offline Extraction
  - Legacy vs. OLTP

There are different types of data extraction which can be broadly categorized into two data extraction techniques, logical and physical. The extraction method you should choose is highly dependent on the source system and also on the business needs in the target data warehouse environment. Very often, there's no possibility to add additional logic to the source systems to enhance an incremental extraction of data due to the performance or the increased workload of these systems. Sometimes even the customer is not allowed to add anything to an out-of-the-box (shrink wrapped CD installable) application system as the performance of the source system has been maximized and carefully calibrated and monitored. The reason being creation of the DWH does not directly help the operational people in any way, for them asking for data every now and then is an annoyance.

The estimated amount of the data to be extracted and the stage in the ETL process (initial load or maintenance of data i.e. data already loaded) may also effect the decision of how to extract, from a logical and a physical perspective. Basically, you have to decide how to extract data logically and physically. Let's discuss logical techniques first in detail.

Logical Data Extraction

- **Full Extraction**
  - The data extracted completely from the source system.
  - No need to keep track of changes.
  - Source data made available as-is w/o any additional information.
- **Incremental Extraction**
  - Data extracted after a well defined point/event in time.
  - Mechanism used to reflect/record the temporal changes in data (column or table).
  - Sometimes entire tables off-loaded from source system into the DWH.
  - Can have significant performance impacts on the data warehouse server.

The two logical data extraction types are full and incremental extraction techniques.
Let's look at the two in detail.

Full Extraction
The data is extracted completely from the source system. Since this extraction reflects all the data currently available in the source system, there's no need to keep track of changes to the data source since the last successful extraction. The source data will be provided as-is and no additional logical information (for example, timestamps etc) is necessary on the source site. An example for a full extraction may be an export file of a distinct table or a remote SQL statement scanning the complete source table.

Incremental Extraction
At a specific point in time, only the data that has changed since a well-defined event back in history will be extracted. This event may be the last time of extraction or a more complex business event like the last day of balancing of accounts. To identify this incremental change there must be a mechanism to identify all the changed information since this specific time event. This information can be either provided by the source data itself like an application column, reflecting the last-changed timestamp or a change table where an appropriate additional

mechanism keeps track of the changes besides the originating transactions. In most cases, using the latter method means adding extraction logic to the source system.

Many data warehouses do not use any change-capture techniques as part of the extraction process. Instead, entire tables from the source systems are extracted to the data warehouse or staging area, and these tables are compared with a previous extract from the source system to identify the changed data. This approach may not have significant impact on the source systems, but it can clearly place considerable burden on the data warehouse processes, particularly if the data volumes are large.

Physical Data Extraction…

- **Online Extraction**
  - Data extracted directly from the source system.
  - May access source tables through an intermediate system.
  - Intermediate system usually similar to the source system.
- **Offline Extraction**
  - Data NOT extracted directly from the source system, instead staged explicitly outside the original source system.
  - Data is either already structured or was created by an extraction routine.
  - Some of the prevalent structures are:
    - Flat files
    - Dump files
    - Redo and archive logs
    - Transportable table-spaces

Physical Extraction Methods

Depending on the chosen logical extraction method and the capabilities and restrictions on the source side, the extracted data can be physically extracted by two mechanisms. The data can either be extracted online from the source system or from an offline structure. Such an offline structure might already exist or it might be generated by an extraction routine.

Online Extraction

The data is extracted directly from the source system itself. The extraction process can connect directly to the source system to access the source tables themselves or to an intermediate system that stores the data in a preconfigured manner (for example, snapshot logs or change tables). Note that the intermediate system is not necessarily physically different from the source system. With online extractions, you need to consider whether the distributed transactions are using original source objects or prepared source objects.

Offline Extraction

The data is not extracted directly from the source system but is staged explicitly outside the original source system. The data already has an existing structure (for example, redo logs, archive logs or transportable table-spaces) or was created by an extraction routine.

You should consider the following structures:

- Flat files
- Data in a defined, generic format. Dump files
- DBMS-specific format. Redo and archive logs
- Transportable table-spaces

Physical Data Extraction

- **Legacy vs. OLTP**
  - Data moved from the source system
  - Copy made of the source system data
  - Staging area used for performance reasons

During extraction, data may be removed from the source system or a copy made and the original data retained in the source system. It is common to move historical data that accumulates in an operational OLTP system to a data warehouse to maintain OLTP performance and efficiency. Legacy systems may require too much effort to implement such offload processes, so legacy data is often copied into the data warehouse, leaving the original data in place. Extracted data is loaded into the data warehouse staging area (a relational database usually separate from the data warehouse database), for manipulation by the remaining ETL processes. Data extraction processes can be implemented using SQL stored procedures, Vendor tools, or custom applications developed in programming or scripting languages.

Data Transformation

- **Basic tasks**
  - Selection
  - Splitting/Joining
  - Conversion
  - Summarization
  - Enrichment

The next operation in the ETL process is the Data Transformation. The major tasks performed during this phase vary depending on the application; however the basic tasks are discussed here.

Selection: This takes place at the beginning of the whole process of data transformation. You select either whole records or parts of several records from the source systems. The task of selection usually forms part of the extraction function itself. However, in some cases, the composition of the source structure may not be supporting selection of the necessary parts during data extraction. In these cases, it is advised to extract the whole record and then do the selection as part of the transformation function.

Splitting/joining: This task includes the types of data manipulation you need to perform on the selected parts of source records. Sometimes (uncommonly), you will be splitting the selected parts even further during data transformation. Joining of parts selected from many source systems is more widespread in the data warehouse environment.

Conversion: This is an all-inclusive task. It includes a large variety of rudimentary conversions of single fields for two primary reasons (i) to standardize among the data extractions from disparate source systems, and (ii) to make the fields usable and understandable to the users.

Summarization: Sometimes you may find that it is not feasible to keep data at the lowest level of detail in your data warehouse. It may be that none of your users ever need data at the lowest granularity for analysis or querying. For example, for a grocery chain, sales data at the lowest level of detail for every transaction at the checkout may not be needed. Storing sales by product by store by day in the data warehouse may be quite adequate. So, in this case, the data transformation function includes summarization of daily sales by product and by store.

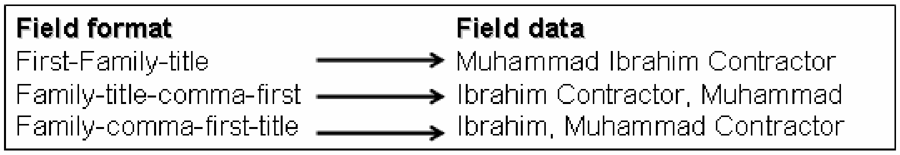
Enrichment: This task is the rearrangement and simplification of individual fields to make them more useful for the data warehouse environment. You may use one or more fields from the same input record to create a better view of the data for the data warehouse. This principle is extended when one or more fields originate from multiple records, resulting in a single field for the data warehouse.

To better understand lets discuss conversion and enrichment with examples.

Data Transformation Basic Tasks: Conversion

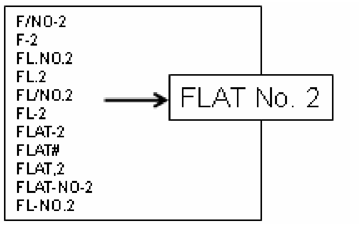
- Convert common data elements into a consistent form i.e. name and address.

Table-16.1 (a)



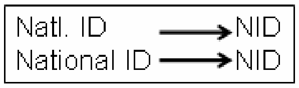
Field format		Field data
First-Family-title	→	Muhammad Ibrahim Contractor
Family-title-comma-first	→	Ibrahim Contractor, Muhammad
Family-comma-first-title	→	Ibrahim, Muhammad Contractor

- Translation of dissimilar codes into a standard code.



F/NO-2
F-2
FL.NO.2
FL.2
FL/NO.2
FL-2
FLAT-2
FLAT#
FLAT,2
FLAT-NO-2
FL-NO.2
→ FLAT No. 2

Table-16.1(b)



Natl. ID	→	NID
National ID	→	NID

Table-16.1: Data Transformation Basic Tasks: Conversion

As discussed earlier, conversion is performed to standardize data by converting data from multiple heterogeneous sources into a single consistent form making it usable and understandable. For example, a data field containing name and job title can be represented in a number of ways in different source systems as shown in Table 16.1(a). Here three different formats have been shown for the same field. Similarly, Table 16.1(b) shows another example of code standardization of dissimilar code representations. Here, two different codes have been shown for representing *National Identity* which can be converted to a consistent form like NID. Same is the case for address representation like house number and flat number etc.

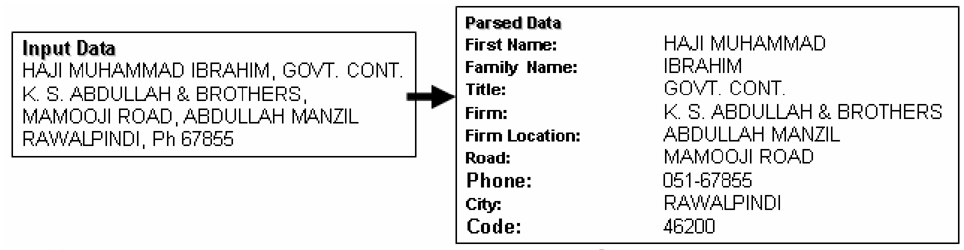
Data Transformation Basic Tasks: Conversion

- Data representation change
  - EBCIDIC to ASCII
- Operating System Change
  - Mainframe (MVS) to UNIX
  - UNIX to NT or XP
- Data type change
  - Program (Excel to Access), database format (FoxPro to Access).
  - Character, numeric and date type.
  - Fixed and variable length.

Why differences in data from different sources? Three common reasons are because data at different locations may have different data representation codes, different operating systems and different data types.

Data Transformation Basic Tasks: Enrichment

- Data elements are mapped from source tables and files to destination fact and dimension tables.


Input Data
HAJI MUHAMMAD IBRAHIM, GOVT. CONT.
K. S. ABDULLAH & BROTHERS,
MAMOOJI ROAD, ABDULLAH MANZIL
RAWALPINDI, Ph 67855

Parsed Data	
First Name:	HAJI MUHAMMAD
Family Name:	IBRAHIM
Title:	GOVT. CONT.
Firm:	K. S. ABDULLAH & BROTHERS
Firm Location:	ABDULLAH MANZIL
Road:	MAMOOJI ROAD
Phone:	051-67855
City:	RAWALPINDI
Code:	46200

(a) **(b)**

- Default values are used in the absence of source data.
- Fields are added for unique keys and time elements.

Figure-16.2: Data Transformation by Enrichment

Enrichment is one of the basic tasks of data transformation as shown in Figure 16.2. Figure -16(a) shows input data that does not have any associated information that links semantics with the given data i.e. what means what? Hover after enrichment the contents of the input data are assigned to the corresponding attributes or fields. Recognize that this could be a very difficult task if the order of the contents of the input varies across the records. Assuming that is not the case, the assignment can be very straightforward. There are certain values that are not given in the input but are implied, such as the postal code, or the phone code etc. In such a case default values are used based on the input data using standard default values. Other information added could be unique keys for identification of data, such that the keys are independent of the business rules, also the data could be time stamped to ascertain its “freshness”.

Significance of Data Loading Strategies

- Need to look at:
  - Data freshness
  - System performance
  - Data volatility
- Data Freshness
  - Very fresh low update efficiency
  - Historical data, high update efficiency
  - Always trade-offs in the light of goals
- System performance
  - Availability of staging table space
  - Impact on query workload
- Data Volat ility
  - Ratio of new to historical data
  - High percentages of data change (batch update)

Once data has been transformed, the *loading* operation is performed. There are different loading strategies and choosing the one depends on data freshness and performance. We also need to look into data volatility, in other words, how much data is changed within the window of the refresh. So in general if real time or near real-time availability of data is required, meaning low update efficiency, because if want the data to be very fresh, then I would not allow accumulation of lot of data, because it means waiting a long time before data is inserted. Whereas, if I want the maximum update efficiency i.e. highest performance, then batch processing would be required. Thus as always there is a tradeoff between performance and data freshness and you need to look at the goals and objectives of your data warehouse, how you follow them and how important these characteristics are. So again there is no one right or wrong answer, depends on what are the goals of the data warehouse, and then you design appropriate to those goals.

The data loading strategy will also depend on the data storage requirements. Depending upon which strategy you use, some require more data storage than others, staging tables etc. You want to look at the impact on query workloads, so when loading the data, at the same time these queries are running, what does that mean? Meaning what is allowable and what is not allowable.

It is also important to look at the ratio of existing to new data? And that will determine or help determine what the strategy should be for implementation. So we need to look at what are the characteristics of workloads that I am interested in. So there are clearly some tradeoffs here. If we look at a loading strategy with a high percentage of data changes per data block, this normally means I am doing a batch update. In other words, I have got a lot of data that I am inserting into the table, and therefore each data block has a lot of rows that are changing per data block.

Three Loading Strategies

- Once we have transformed data, there are three primary loading strategies:
- Full data refresh with BLOCK INSERT or ‘block slamming’ into empty table.
- Incremental data refresh with BLOCK INSERT or ‘block slamming’ into existing (populated) tables.
- Trickle/continuous feed with constant data collection and loading using row level insert and update operations.

There can be a couple of lectures on loading strategies, but in this course will limit our selves to the basic concepts only. Once the data has been transformed, it is ready to be loaded into the DWH, for this purpose three loading strategies are prevalent. When the tables are populated for the first time it is a full data refresh, depending on the tool or the environment being used, this may be called as BLOCK INSERT or “block slamming” i.e. large blocks of data are loaded, usually in the form of batch updates. As the DWH evolves over time, there may be no empty tables, but already filled table have to be updated. For this purpose DWH has to be refreshed incrementally. Depending on the line of business and other reporting requirements the refresh period will vary and also the amount of data to be loaded. In extreme cases this may boil down to BLOCK INSERT or “block slamming” into the main DWH tables. This will have certain performance implications with reference to availability of data, and the outcome of the queries while the data is being refreshed and requirement of additional memory space in the form of “shadow tables”, but the discussion is beyond the scope of this course. The third and final loading strategy is trickle feed. This strategy is more in the context of active data warehousing and has a high update overhead. Usually the criterion when to load depends on the time interval between uploads or the amount of data recorded or their combination.

Lecture-17
Issues of ETL

Why ETL Issues?

Data from different source systems will be different, poorly documented and dirty. Lot of analysis required.

Easy to collate addresses and names? Not really. No address or name standards.

Use software for standardization. Very expensive, as any “standards” vary from country to country, not large enough market.

It is very difficult to extract the data from different source systems, because by definition they are heterogeneous, and as a consequence, there are multiple representations of the same data, have inconsistent data formats, have very poor documentation and have dirty data etc. Therefore, you have to figure different ways of solving this problem, requiring lot of analysis.

Consider the deceptively simple task of automating the collating of addresses and names from different operational databases. You can’t predict or legislate format or content, therefore, there are no address standards, and there are no standard address and name cleaning software.

There is software for certain geographies, but the addresses and names in US will be completely different from that in Austria or Brazil. Every country finds a way of writing the addresses differently, that is always a big problem. Hence automation does not solve the problem, and also has a high price tag in case of 1st generation ETL tools ($200K-$400K range).

Why ETL Issues?

Things would have been simpler in the presence of operational systems, but that is not always the case

Manual data collection and entry. Nothing wrong with that, but potential to introduces lots of problems.

Data is never perfect. The cost of perfection, extremely high vs. its value.

Most organizations have ERP systems (which are basically operational systems) from where they load their DWH. However, this is not a rule, but used in an environment where transaction processing systems are in place, and the data is generated in real time.

However, in the case of a country-wide census, there are no source systems, so data is being collected directly and manually. People go form door to door and get forms filled, subsequently the data in its most raw form is manually entered into the database. There is nothing wrong with that, the problem being human involvement in data recording and storage is very expensive, time consuming and prone to errors. There have to be control and processes to colle ct as clean a data as possible.

People have a utopian view that they will collect perfect data. This is wrong. The cost of collecting perfect data far exceeds its value. Therefore, the objective should be to get as clean data as can be based on the given constraints.

"Some" Issues

- Usually, if not always underestimated
- Diversity in source systems and platforms
- Inconsistent data representations
- Complexity of transformations
- Rigidity and unavailability of legacy systems
- Volume of legacy data
- Web scrapping

ETL is something which is usually always underestimated; consequently there are many many issues. There can be an entire course on ETL. Since this course is not only about ETL, therefore, we will limit our self to discussing only some of the issues. We will now discuss some of the more important issues one-by-one.

Complexity of problem/work underestimated

- Work seems to be deceptively simple.
- People start manually building the DWH.
- Programmers underestimate the task.
- Impressions could be deceiving.
- Traditional DBMS rules and concepts break down for very large heterogeneous historical databases.

At first glance, when data is moved from the legacy environment to the data warehouse environment, it appears there is nothing more going on than simple extraction of data from one place to the next. Because of the deceptive simplicity, many organizations start to build their data warehouse manually. The programmer looks at the movement of data from the old operational environment to the new data warehouse environment and declares "I can do that!" With pencil and writing pad in hand the programmer anxiously jumps into the creation of code in the first three minutes of the design and development of the data warehouse.

However, first impressions can be deceiving - in this case very deceiving. What at first appears to be nothing more than the movement of data from one place to another quickly turns into a large and complex task, actually a nightmare. The scope of the problem being far larger and far more complex than the programmer had ever imagined.

Diversity in source systems and platforms

Platform	OS	DBMS	MIS/ERP
Main Frame	VMS	Oracle	SAP
Mini Computer	Unix	Informix	PeopleSoft
Desktop	Win NT	Access	JD Edwards
	DOS	Text file	

Dozens of source systems across organizations

Numerous source systems within an organization

Need specialist for each

Table-17.1: Diversity in source systems and platforms

There are literally more than a dozen different source system technologies in use in the market today. Example of the kind of source system technologies that you often see are shown in Table 17.1. The team doing ETL processing in any given organization probably has to understand at least 6 of them. You will be surprised to find big banks and telecommunication companies that have every single one of these technologies; in a single company. It is a nightmare to understand them. The only viable solution is to have a technical expert that knows how to get data from every single one of these formats. In short it's a nightmare.

Same data, different representation

Date value representations
Examples:

970314	1997-03-14
03/14/1997	14-MAR-1997
March 14 1997	2450521.5 (Julian date format)

Gender value representations
Examples:

- Male/Female	- M/F
- 0/1	- PM/PF

There are 127 different ways to spell AT&T; there are 1000 ways to spell duPont. How many ways are there to represent date? What is the Julian date format? All of these questions and many more are at the core of inconsistent data representation. Don't be surprised to find source system with Julian date format or dates stored as text strings.

Consider the case of gender representation. You might think it is just male and female. Wrong. It's not just male and female, it also includes unknown, because for some people you don't have their gender data. If this was not sufficient, there is another twist to the gender i.e. instead of M and F, you come across PM and PF. What is this? This is probable male and probable female i.e. based on the name the gender has been guessed. For example, if the name is Mussarat it is probable female. It is not for sure, but we think it is. If it's Riffat, it is probable male.

So you get all these weird and strange representations in operational data, while in a DWH there has to be only ONE and consistent representation. You just can't dump all the operational data independent of which source system it came from that has a different data representation. Remember in the data model you should have non-overlapping consistent domains for every attribute.

Multiple sources for same data element

Need to rank source systems on a per data element basis.

Take data element from source system with highest rank wh ere element exists.

"Guessing" gender from name

Something is better than nothing?

Must sometimes establish "group ranking" rules to maintain data integrity.

First, middle and family name from two systems of different rank. People using middle name as first name.

This means you need to establish ranking in the source system on per element (attribute) basis. Ranking is all about selecting the "right" source system. Rank establishment has to be based on which source system is known to have the cleanest data for a particular attribute. Obviously you take the data element from the source system with the highest rank where the element exists. However, you have to be clever about how you use the rank.

For example, consider the case of the gender data coming from two different source systems A and B. It may be the case that the highest quality data is from source system A, where the boxes for the gender were checked by the customers themselves. But what if someone did not check the gender box? Then you go on to the next cleanest source system i.e. B, where the gender was guessed based on the name.

Obviously the quality of available data for source system B is not as good as that of source system A, but since you do not have data for this particular indiv idual in source systems A, so it is better to have something then nothing. This is arguable i.e. maybe it is better to have nothing than to have dirty data. The point is if you have some level of confidence in the data you should take it. Typically it is a good idea to remember from where you took the data, so you can use this information from an analytic point of view e.g. where you get clean data etc.

Consider the case of name i.e. first name, middle name and family name and two source systems i.e. C and D. Assume that source system C has higher quality data entry, and management and processes and controls as compared to D. Also assume that source system C does not have the middle name, while source system D has the complete name. Since C has a higher precedence, so you take the first name and the family name from it. But you really would like to have the middle name so you take it from source system D. This turns out to be a big problem, because people sometimes use their middle name as their first name e.g. *Imran Baqai* instead of *Shezad Imran Baqai*.

Complexity of required transformations

Simple one-to-one scalar transformations
- 0/1 ? M/F

One-to-many element transformations
- 4 x 20 address field ? House/Flat, Road/Street, Area/Sector, City.

Many-to-many element transformations
- House-holding (who live together) and individualization (who are same) and same lands.

There is a spectrum of simple all the way up to very complex transformations that you can implement. And you probably end up with many in most DWH deployments. The transformations are typically divided into three categories as follows:

- Simple one -to-one scalar transformations.
- One -to-many element transformations.
- Complex many-to-many element transformations.

Simple scalar transformation is a one-to-one mapping from one set of values to another set of values using straightforward rules. For example, if 0/1 corresponds to male/female in one source system, then the gender is stored as male/female in the DW.

A one-to-many transformation is more complex than scalar transformation. As a data element form the source system results in several columns in the DW. Consider the 6×30 address field (6 lines of 30 characters each), the requirement is to parse it into street address lines 1 and 2, city, sate and zip code by applying a parsing algorithm.

The most complex is many-to-many element transformations. Good examples are house holding and individualization. This is achieved by using candidate keys and fuzzy matching to determine which individuals are the same individuals, and which individuals go in the same household and so on. This is a very complex transformation and will be discussed in BSN lecture.

Rigidity and unavailability of legacy systems

- Very difficult to add logic to or increase performance of legacy systems.
- Utilization of expensive legacy systems is optimized.
- Therefore, want to off-load transformation cycles to open systems environment.
- This often requires new skill sets.
- Need efficient and easy way to deal with incompatible mainframe data formats.

Legacy systems typically mean mainframe type systems that are very expensive to buy and operate. We are talking about CPU cycles as well as storage, but mainly computing CPU cycles. The mainframe machines are usually 110% utilized in most businesses. The reason being, businesses want the “most bang for the buck”, so they keep on putting more and more load on the machines.

Consider an expensive ERP system, when is the system available for the processing of DWH transformations? Probably never, as the system utilization has already been maxed up. So the big question is where to do the transformations? Where do the CPU cycles come from? And when the legacy data is in a mainframe environment, typically the cost of transformation is very high because of scarcity and cost of legacy cycles. But you don't want the cost of constructing the DWH to exceed the value of the DWH. So you need to think of ways to move the cycles to more cost efficient environments.

Therefore, in case of legacy environment, generally you want to off-load the transformation cycles to open systems environment. Open system i.e. Intel chipset, operating systems like Unix and NT etc. as it results in a much more cost effective approach than doing things in a mainframe type of environment. However, this may sometimes be problematic for some organizations requiring certain new skill sets. For example the IT department of a bank may have only used a mainframe for day-to-day operations, and moving on to open systems may be painful for them due to the learning curve etc.

It is NOT simply moving the data to open systems; there are lots of issues due to legacy systems, simple as well as complex. For example, in the mainframe environment you have EBCDIC character representation and in open systems it is always ASCII, so you have to do the conversion. In mainframe environment there is packed decimal encoding, zone decimal and weird date formats that are not understandable in the open systems environment. So you have to have utilities for doing the conversion and so on before the required transformation takes place.

Volume of legacy data

- Talking about not weekly data, but data spread over years.
- Historical data on tapes that are serial and very slow to mount etc.
- Need lots of processing and I/O to effectively handle large data volumes.
- Need efficient interconnect bandwidth to transfer large amounts of data from legacy sources to DWH.

The volume of legacy data is typically very large. It is not just going back and getting tape for one month of billing data, but getting billing data for as much as one could get hold of (say), three years of billing data. Imagine the volume of work, lots of processing and I/Os especially when the data is on tapes. This would involve tape mounts, and manual tape mounts is painstakingly slow. It takes forever to mount hundreds and hundreds of tapes. The IT people will hate you after you are through with tape mounts and data retrieval.

Assuming transformation takes place on the mainframe; the next issue is to move large volumes of data from the mainframe into the DWH requiring network bandwidth. The data may have to be moved across the LAN, and maybe even across the WAN which is more painful. Therefore, the DWH architects need to pay a lot of attention to capacity planning issues of the DWH i.e. how to size the DWH, how much is it going to cost to extract all the data, prepare the data, and move the data from a network capacity planning perspective. Ensure that from a capacity planning point of view all these aspects are considered.

Web scrapping

- Lot of data in a web page, but is mixed with a lot of "junk".
- Problems:
  - Limited query interfaces
    - Fill in forms
  - "Free text" fields
    - E.g. addresses
  - Inconsistent output
    - i.e., html tags which mark interesting fields might be different on different pages.
  - Rapid change without notice.

During the last decade the web has grown from a curiosity to a required information repository. From news headlines and stories, financial data and reports, entertainment events, online shopping and business to business commerce, the Web has it all. The promise of the web is that all of this information will be easily maintained, monitored, queried, and browsed. Some of the promise has already been realized but much remains to be done. Web browsers (such as Internet Explorer and Netscape) have been very successful at displaying web-based information on a computer monitor.

However, there remains a divide between content presented on web pages and the ability to extract that content and make it actionable. What companies and individuals need is a method to extract or mine only that data or content in which they are interested. Furthermore, they need to be able to take that mined content, apply different kinds of data processing, and make it available for use in different applications, and have it delivered to a database, email, spreadsheet, mobile phone, or whatever delivery destination device is available.

Web scrapping is a process of applying screen scrapping techniques to the web. There are several web scrapping products in the market and target business users who want to creatively use the data, not write complex scripts. Some of the uses of scrapping are:

- Building contact lists
- Extracting product catalogs
- Aggregating real-estate info
- Automating search Ad listings
- Clipping news articles etc.

Beware of data quality (or lack of it)

- Data quality is always worse than expected.
- Will have a couple of lectures on data quality and its management.
- It is not a matter of few hundred rows.
- Data recorded for running operations is not usually good enough for decision support.
    - Correct totals don't guarantee data quality.
    - Not knowing gender does not hurt POS.
    - Centurion customers popping up.

Personal word of warning, the data quality will be always be worse than you expect, you can count on this. Everybody says that other people have dirty data (like everyone thinks they will win the lottery and other people will have an accident), but my d ata is clean. I don't believe you; your data is always dirty. You need to allocate time and resources to facilitate the data cleanup. In the next series of lectures, we will talk about a methodology using TQM techniques and apply to DWH data quality. There is a whole methodology on how to deliver data quality in a data warehouse environment. This is just a warning, data quality is always worse than you think it is. You have used data for transaction processing; it does not mean it is good for decision suppo rt.

Everybody always underestimate how dirty their data is. It is not a few hundred rows of dirty data. Data is always dirtier than you think. I have not seen a single business environment that has data that is as clean as it should be. Especially those things required for decision making. Most people concentrate on data quality on numbers, making sure the amounts are right, make sure the accounts are right those kinds of things because you need those to print statements to do business. But for decision making I need different things. I need to know the gender of the customer to understand my market place. I don't need to know the gender of the customer to process a transaction. So basically nobody cares about it, and they put any garbage they want in there and that becomes a problem later on. Absolutely do not assume the data is clean, assume the opposite, assume the data is dirty and help prove it otherwise.

ETL vs. ELT

There are two fundamental approaches to data acquisition:

ETL: Extract, Transform, Load in which data transformation takes place on a separate transformation server.

ELT: Extract, Load, Transform in which data transformation takes place on the data warehouse server.

Combination of both is also possible

You want to have a data flow driven architecture that you understand these points i.e. what data is required and how does it flow in the system driven by the meta data. Once we have those data flows then we can parallelize. You leverage pre -packaged tool for the transformation steps whenever possible. But then again have to look at the practical realities of the market place. That's why I say whenever possible. Whenever possible partially means whenever economically feasible.

In the architecture, ETL presents itself very suitable for data parallelism. Because I have got tons of data and I want to apply operations consistently across all of that data. So data parallelism is almost always used and you may use pipeline parallelism assuming I do not need too many sources so on. The difficulty of doing this parallelization by hand is very high. Because if you want to write a parallel program say a parallel C program, like parallel pro or something like that, it is very difficult. Again it is not just writing a parallel program, but you a lso need to make sure it works with check-points restarts and/or error conditions and all those things. I suspect that in your environment, but this is true for NADRA and PTML, that use the database to do the parallelization. So a different kind of approach, what if you are not willing to buy a tool that costs US$ 200,000?

This is a different kind of approach called ELT which is Extract Load Transform. We extract, we load into the database and then we transform in the parallel database. Then we get all the parallelism for free, because you already have a parallel database. You don't have to buy a separate tool in order to get the parallelization.

Lecture-17
Issues of ETL

Why ETL Issues?

Data from different source systems will be different, poorly documented and dirty. Lot of analysis required.

Easy to collate addresses and names? Not really. No address or name standards.

Use software for standardization. Very expensive, as any "standards" vary from country to country, not large enough market.

It is very difficult to extract the data from different source systems, because by definition they are heterogeneous, and as a consequence, there are multiple representations of the same data, have inconsistent data formats, have very poor documentation and have dirty data etc. Therefore, you have to figure different ways of solving this problem, requiring lot of analysis.

Consider the deceptively simple task of automating the collating of addresses and names from different operational databases. You can't predict or legislate format or content, therefore, there are no address standards, and there are no standard address and name cleaning software.

There is software for certain geographies, but the addresses and names in US will be completely different from that in Austria or Brazil. Every country finds a way of writing the addresses differently, that is always a big problem. Hence automation does not solve the problem, and also has a high price tag in case of 1st generation ETL tools ($200K-$400K range).

Why ETL Issues?

Things would have been simpler in the presence of operational systems, but that is not always the case

Manual data collection and entry. Nothing wrong with that, but potential to introduces lots of problems.

Data is never perfect. The cost of perfection, extremely high vs. its value.

Most organizations have ERP systems (which are basically operational systems) from where they load their DWH. However, this is not a rule, but used in an environment where transaction processing systems are in place, and the data is generated in real time.

However, in the case of a country-wide census, there are no source systems, so data is being collected directly and manually. People go form door to door and get forms filled, subsequently the data in its most raw form is manually entered into the database. There is nothing wrong with that, the problem being human involvement in data recording and storage is very expensive, time consuming and prone to errors. There have to be control and processes to collect as clean a data as possible.

People have a utopian view that they will collect perfect data. This is wrong. The cost of collecting perfect data far exceeds its value. Therefore, the objective should be to get as clean data as can be based on the given constraints.

"Some" Issues

- Usually, if not always underestimated
- Diversity in source systems and platforms
- Inconsistent data representations
- Complexity of transformations
- Rigidity and unavailability of legacy systems
- Volume of legacy data
- Web scrapping

ETL is something which is usually always underestimated; consequently there are many many issues. There can be an entire course on ETL. Since this course is not only about ETL, therefore, we will limit our self to discussing only some of the issues. We will now discuss some of the more important issues one-by-one.

Complexity of problem/work underestimated

- Work seems to be deceptively simple.
- People start manually building the DWH.
- Programmers underestimate the task.
- Impressions could be deceiving.
- Traditional DBMS rules and concepts break down for very large heterogeneous historical databases.

At first glance, when data is moved from the legacy environment to the data warehouse environment, it appears there is nothing more going on than simple extraction of data from one place to the next. Because of the deceptive simplicity, many organizations start to build their data warehouse manually. The programmer looks at the movement of data from the old operational environment to the new data warehouse environment and declares "I can do that!" With pencil and writing pad in hand the programmer anxiously jumps into the creation of code in the first three minutes of the design and development of the data warehouse.

However, first impressions can be deceiving - in this case very deceiving. What at first appears to be nothing more than the movement of data from one place to another quickly turns into a large and complex task, actually a nightmare. The scope of the problem being far larger and far more complex than the programmer had ever imagined.

Diversity in source systems and platforms

Platform	OS	DBMS	MIS/ERP
Main Frame	VMS	Oracle	SAP
Mini Computer	Unix	Informix	PeopleSoft
Desktop	Win NT	Access	JD Edwards
	DOS	Text file	

Dozens of source systems across organizations

Numerous source systems within an organization

Need specialist for each

Table-17.1: Diversity in source systems and platforms

There are literally more than a dozen different source system technologies in use in the market today. Example of the kind of source system technologies that you often see are shown in Table 17.1. The team doing ETL processing in any given organization probably has to understand at least 6 of them. You will be surprised to find big banks and telecommunication companies that have every single one of these technologies; in a single company. It is a nightmare to understand them. The only viable solution is to have a technical expert that knows how to get data from every single one of these formats. In short it's a nightmare.

Same data, different representation

Date value representations
Examples:

970314	1997-03-14
03/14/1997	14-MAR-1997
March 14 1997	2450521.5 (Julian date format)

Gender value representations
Examples:

- Male/Female	- M/F
- 0/1	- PM/PF

There are 127 different ways to spell AT&T; there are 1000 ways to spell duPont. How many ways are there to represent date? What is the Julian date format? All of these questions and many more are at the core of inconsistent data representation. Don't be surprised to find source system with Julian date format or dates stored as text strings.

Consider the case of gender representation. You might think it is just male and female. Wrong. It's not just male and female, it also includes unknown, because for some people you don't have their gender data. If this was not sufficient, there is another twist to the gender i.e. instead of M and F, you come across PM and PF. What is this? This is probable male and probable female i.e. based on the name the gender has been guessed. For example, if the name is Mussarat it is probable female. It is not for sure, but we think it is. If it's Riffat, it is probable male.

So you get all these weird and strange representations in operational data, while in a DWH there has to be only ONE and consistent representation. You just can't dump all the operational data independent of which source system it came from that has a different data representation. Remember in the data model you should have non-overlapping consistent domains for every attribute.

Multiple sources for same data element

Need to rank source systems on a per data element basis.

Take data element from source system with highest rank where element exists.

"Guessing" gender from name

Something is better than nothing?

Must sometimes establish "group ranking" rules to maintain data integrity.

First, middle and family name from two systems of different rank. People using middle name as first name.

This means you need to establish ranking in the source system on per element (attribute) basis. Ranking is all about selecting the "right" source system. Rank establishment has to be based on which source system is known to have the cleanest data for a particular attribute. Obviously you take the data element from the source system with the highest rank where the element exists. However, you have to be clever about how you use the rank.

For example, consider the case of the gender data coming from two different source systems A and B. It may be the case that the highest quality data is from source system A, where the boxes for the gender were checked by the customers themselves. But what if someone did not check the gender box? Then you go on to the next cleanest source system i.e. B, where the gender was guessed based on the name.

Obviously the quality of available data for source system B is not as good as that of source system A, but since you do not have data for this particular individual in source systems A, so it is better to have something then nothing. This is arguable i.e. maybe it is better to have nothing than to have dirty data. The point is if you have some level of confidence in the data you should take it. Typically it is a good idea to remember from where you took the data, so you can use this information from an analytic point of view e.g. where you get clean data etc.

Consider the case of name i.e. first name, middle name and family name and two source systems i.e. C and D. Assume that source system C has higher quality data entry, and management and processes and controls as compared to D. Also assume that source system C does not have the middle name, while source system D has the complete name. Since C has a higher precedence, so you take the first name and the family name from it. But you really would like to have the middle name so you take it from source system D. This turns out to be a big problem, because people sometimes use their middle name as their first name e.g. *Imran Baqai* instead of *Shezad Imran Baqai*.

Complexity of required transformations

Simple one-to-one scalar transformations
- 0/1 ? M/F

One-to-many element transformations
- 4 x 20 address field ? House/Flat, Road/Street, Area/Sector, City.

Many-to-many element transformations
- House-holding (who live together) and individualization (who are same) and same lands.

There is a spectrum of simple all the way up to very complex transformations that you can implement. And you probably end up with many in most DWH deployments. The transformations are typically divided into three categories as follows:

- Simple one -to-one scalar transformations.
- One -to-many element transformations.
- Complex many-to-many element transformations.

Simple scalar transformation is a one-to-one mapping from one set of values to another set of values using straightforward rules. For example, if 0/1 corresponds to male/female in one source system, then the gender is stored as male/female in the DW.

A one-to-many transformation is more complex than scalar transformation. As a data element form the source system results in several columns in the DW. Consider the 6×30 address field (6 lines of 30 characters each), the requirement is to parse it into street address lines 1 and 2, city, sate and zip code by applying a parsing algorithm.

The most complex is many-to-many element transformations. Good examples are house holding and individualization. This is achieved by using candidate keys and fuzzy matching to determine which individuals are the same individuals, and which individuals go in the same household and so on. This is a very complex transformation and will be discussed in BSN lecture.

Rigidity and unavailability of legacy systems

- Very difficult to add logic to or increase performance of legacy systems.
- Utilization of expensive legacy systems is optimized.
- Therefore, want to off-load transformation cycles to open systems environment.
- This often requires new skill sets.
- Need efficient and easy way to deal with incompatible mainframe data formats.

Legacy systems typically mean mainframe type systems that are very expensive to buy and operate. We are talking about CPU cycles as well as storage, but mainly computing CPU cycles. The mainframe machines are usually 110% utilized in most businesses. The reason being, businesses want the “most bang for the buck”, so they keep on putting more and more load on the machines.

Consider an expensive ERP system, when is the system available for the processing of DWH transformations? Probably never, as the system utilization has already been maxed up. So the big question is where to do the transformations? Where do the CPU cycles come from? And when the legacy data is in a mainframe environment, typically the cost of transformation is very high because of scarcity and cost of legacy cycles. But you don't want the cost of constructing the DWH to exceed the value of the DWH. So you need to think of ways to move the cycles to more cost efficient environments.

Therefore, in case of legacy environment, generally you want to off-load the transformation cycles to open systems environment. Open system i.e. Intel chipset, operating systems like Unix and NT etc. as it results in a much more cost effective approach than doing things in a mainframe type of environment. However, this may sometimes be problematic for some organizations requiring certain new skill sets. For example the IT department of a bank may have only used a mainframe for day-to-day operations, and moving on to open systems may be painful for them due to the learning curve etc.

It is NOT simply moving the data to open systems; there are lots of issues due to legacy systems, simple as well as complex. For example, in the mainframe environment you have EBCDIC character representation and in open systems it is always ASCII, so you have to do the conversion. In mainframe environment there is packed decimal encoding, zone decimal and weird date formats that are not understandable in the open systems environment. So you have to have utilities for doing the conversion and so on before the required transformation takes place.

Volume of legacy data

- Talking about not weekly data, but data spread over years.
- Historical data on tapes that are serial and very slow to mount etc.
- Need lots of processing and I/O to effectively handle large data volumes.
- Need efficient interconnect bandwidth to transfer large amounts of data from legacy sources to DWH.

The volume of legacy data is typically very large. It is not just going back and getting tape for one month of billing data, but getting billing data for as much as one could get hold of (say), three years of billing data. Imagine the volume of work, lots of processing and I/Os especially when the data is on tapes. This would involve tape mounts, and manual tape mounts is painstakingly slow. It takes forever to mount hundreds and hundreds of tapes. The IT people will hate you after you are through with tape mounts and data retrieval.

Assuming transformation takes place on the mainframe; the next issue is to move large volumes of data from the mainframe into the DWH requiring network bandwidth. The data may have to be moved across the LAN, and maybe even across the WAN which is more painful. Therefore, the DWH architects need to pay a lot of attention to capacity planning issues of the DWH i.e. how to size the DWH, how much is it going to cost to extract all the data, prepare the data, and move the data from a network capacity planning perspective. Ensure that from a capacity planning point of view all these aspects are considered.

Web scrapping

- Lot of data in a web page, but is mixed with a lot of “junk”.
- Problems:
    - Limited query interfaces
        - Fill in forms
    - “Free text” fields
        - E.g. addresses
    - Inconsistent output
        - i.e., html tags which mark interesting fields might be different on different pages.
    - Rapid change without notice.

During the last decade the web has grown from a curiosity to a required information repository. From news headlines and stories, financial data and reports, entertainment events, online shopping and business to business commerce, the Web has it all. The promise of the web is that all of this information will be easily maintained, monitored, queried, and browsed. Some of the promise has already been realized but much remains to be done. Web browsers (such as Internet Explorer and Netscape) have been very successful at displaying web-based information on a computer monitor.

However, there remains a divide between content presented on web pages and the ability to extract that content and make it actionable. What companies and individuals need is a method to extract or mine only that data or content in which they are interested. Furthermore, they need to be able to take that mined content, apply different kinds of data processing, and make it available for use in different applications, and have it delivered to a database, email, spreadsheet, mobile phone, or whatever delivery destination device is available.

Web scrapping is a process of applying screen scrapping techniques to the web. There are several web scrapping products in the market and target business users who want to creatively use the data, not write complex scripts. Some of the uses of scrapping are:

- Building contact lists
- Extracting product catalogs
- Aggregating real-estate info
- Automating search Ad listings
- Clipping news articles etc.

Beware of data quality (or lack of it)

- Data quality is always worse than expected.
- Will have a couple of le ctures on data quality and its management.
- It is not a matter of few hundred rows.
- Data recorded for running operations is not usually good enough for decision support.
    - Correct totals don't guarantee data quality.
    - Not knowing gender does not hurt POS.
    - Centurion customers popping up.

Personal word of warning, the data quality will be always be worse than you expect, you can count on this. Everybody says that other people have dirty data (like everyone thinks they will win the lottery and other peop le will have an accident), but my data is clean. I don't believe you; your data is always dirty. You need to allocate time and resources to facilitate the data cleanup. In the next series of lectures, we will talk about a methodology using TQM techniques a nd apply to DWH data quality. There is a whole methodology on how to deliver data quality in a data warehouse environment. This is just a warning, data quality is always worse than you think it is. You have used data for transaction processing; it does not mean it is good for decision support.

Everybody always underestimate how dirty their data is. It is not a few hundred rows of dirty data. Data is always dirtier than you think. I have not seen a single business environment that has data that is as clean as it should be. Especially those things required for decision making. Most people concentrate on data quality on numbers, making sure the amounts are right, make sure the accounts are right those kinds of things because you need those to print statements to do business. But for decision making I need different things. I need to know the gender of the customer to understand my market place. I don't need to know the gender of the customer to process a transaction. So basically nobody cares about it, and they put any garbage they want in there and that becomes a problem later on. Absolutely do not assume the data is clean, assume the opposite, assume the data is dirty and help prove it otherwise.

ETL vs. ELT

There are two fundamental approaches to data acquisition:

ETL: Extract, Transform, Load in which data transformation takes place on a separate transformation server.

ELT: Extract, Load, Transform in which data transformation takes place on the data warehouse server.

Combination of both is also possible

You want to have a data flow driven architecture that you understand these points i.e. what data is required and how does it flow in the system driven by the meta data. Once we have those data flows then we can parallelize. You leverage pre -packaged tool for the transformation steps whenever possible. But then again have to look at the practical realities of the market place. That's why I say whenever possible. Whenever possible partially means whenever economically feasible.

In the architecture, ETL presents itself very suitable for data parallelism. Because I have got tons of data and I want to apply operations consistently across all of that data. So data parallelism is almost always used and you may use pipeline parallelism assuming I do not need too many sources so on. The difficulty of doing this parallelization by hand is very high. Because if you want to write a parallel program say a parallel C program, like parallel pro or something like that, it is very difficult. Again it is not just writing a parallel program, but you also need to make sure it works with check-points restarts and/or error conditions and all those things. I suspect that in your environment, but this is true for NADRA and PTML, that use the database to do the parallelization. So a different kind of approach, what if you are not willing to buy a tool that costs US$ 200,000?

This is a different kind of approach called ELT which is Extract Load Transform. We extract, we load into the database and then we transform in the parallel database. Then we get all the parallelism for free, because you already have a parallel database. You don't have to buy a separate tool in order to get the parallelization.

Lecture-19
ETL Detail: Data Cleansing

Background

- Other names: Called as data scrubbing or cleaning.
- More than data arranging.
- Big problem, big effect: Enormous problem, as most data is dirty. GIGO
- Dirty is relative.
- Paradox: Must involve domain expert, as detailed domain knowledge is required, so it becomes semi-automatic, but has to be automatic because of large data sets.
- Data duplication.

It is crucial to the process of loading the data warehouse that you not just rearrange the grouping of source data and deliver it to a destination database, but that you also ensure the quality of that data. Data cleansing is vitally important to the overall health of your warehouse project and ultimately affects the health of your company. Do not take this statement lightly.

The true scope of a data cleansing project is enormous. Much of production data is dirty, and you don't even want to consider what work cleaning it up would take. By "dirty," I mean that it does not conform to proper domain definitions or "make sense." The age-old adage "garbage in, garbage out" still applies, and you can do nothing about it short of analyzing and correcting the corporate data. What is noise in one domain may be information in another.

Usually the process of data cleansing can not be performed without the involvement of a domain expert because the detection and correction of anomalies requires detailed domain knowledge. Data cleansing is therefore described as semi-automatic but it should be as automatic as possible because of the large amount of data that usually is be processed and the time required for an expert to cleanse it manually.

The original aim of data cleansing was to eliminate duplicates in a data collection, a problem occurring already in single database applications and gets worse when integrating data from different sources. Will have a complete lecture on it.

Lighter Side of Dirty Data

- Year of birth 1995 current year 2005
- Born in 1986 hired in 1985
- Who would take it seriously? Computers while summarizing, aggregating, populating etc.
- Small discrepancies become irrelevant for large averages, but what about sums, medians, maximum, minimum etc.?

Value validation is the process of ensuring that each value that is sent to the data warehouse is accurate. You may had that experience in which you look at the contents of one of your major flat files or database tables and intuitively pick that the data is incorrect. No way could that employee be born in 2004! You know your company doesn't hire infants. You may also discover another incorrect record. How could someone be born in 1978 but hired in 1977?

All too often, these types of data integrity problems are laughed at and then ignored. All too often people say "No one would actually take that information seriously, would they?" Well, maybe people won't, but MIS systems will. That information can be summarized, aggregated, and/or manipulated in some way, and then populated into another data element. And when that data element is moved into the DWH, analytical processing will be performed on it that can affect the way your company does business. What if data from the DWH is being analyzed to revise hiring practices? That data may make a wrong impact on the business decisions if enough of the hire and birth dates are inaccurate.

Small data discrepancies can become statistically irrelevant when large volumes of data are averaged. But averaging is not the only analytical function that is used in analytical data warehouse queries. What about sums, medians, max/min, and other aggregate and scalar functions? Even further, can you actually prove that the scope of your data problems is as small as you think it is? The answer is probably "no."

Serious Problems due to dirty data

- Decisions taken at government level using wrong data resulting in undesirable results.
- In direct mail marketing sending letters to wrong addresses loss of money and bad reputation.

Administration: The government analyses data collected by population census to decide which regions of the country require further investments in health, education, clean drinking water, electricity etc. because of current and expected future trends. If the rate of birth in one region has increased over the last couple of years, the existing health facilities and doctors employed might not be sufficient to handle the number of current and expected patients. Thus, additional dispensaries or employment of doctors will be needed. Inaccuracies in analyzed data can lead to false conclusions and misdirected release of funds with catastrophic results for a poor country like Pakistan.

Supporting business processes: Erroneous data leads to unnecessary costs and probably bad reputation when used to support business processes. Consider a company using a list of consumer addresses and buying habits and preferences to advertise a new product by direct mailing. Invalid addresses cause the letters to be returned as undeliverable. People being duplicated in the mailing list account for multiple letters sent to the same person, leading to unnecessary expenses and frustration. Inaccurate information about consumer buying habits and preferences contaminate and falsify the target group, resulting in advertisement of products that do not correspond to consumer’s needs. Companies trading such data face the possibility of an additional loss of reputation in case of erroneous data.

3 Classes of Anomalies…

- Syntactically Dirty Data
  - Lexical Errors
  - Irregularities
- Semantically Dirty Data
  - Integrity Constraint Violation
  - Business rule contradiction
  - Duplication
- Coverage Anomalies
  - Missing Attributes
  - Missing Records

Syntactically Dirty Data

Lexical errors: For example, assume the data to be stored in table form with each row representing a tuple and each column an attribute. If we expect the table to have five columns because each tuple has five attributes but some or all of the rows contain only four columns then the actual structure of the data does not conform to the specified format.

Irregularities are concerned with the non-uniform use of values, units and abbreviations.
This can happen for example if we use different currencies to specify an employee's salary. This is especially profound if the currency is not explicitly listed with each value, and is assumed to be uniform. Annual salary or 20,000 means $20,000 or Rs. 20,000. This results in values being correct representations of facts if we have the necessary knowledge about their representation needed to interpret them.

Semantically dirty data

Integrity constraint violations describe tuples (or sets of tuples) that do not satisfy one or more of the integrity constraints. Integrity constraints are used to describe our understanding of the mini-world by restricting the set of valid instances. Each constraint is a rule representing knowledge about the domain and the values allowed for representing certain facts.

Contradictions are values within one tuple or between tuples that violate some kind of dependency between the values. An example for the first case could be a contradiction between the attribute AGE and DATE_OF_BIRTH for a tuple representing persons. Contradictions are either violations of functional dependencies that can be represented as integrity constraints or duplicates with inexact values. They are therefore not regarded as separate data anomaly.

Duplicates are two or more tuples representing the same entity from the mini-world. The values of these tuples do not need to be completely identical. Inexact duplicates are specific cases of contradiction between two or more tuples. They represent the same entity but with different values for all or some of its properties. This hardens the detection of duplicates and there mergence.

3 Classes of Anomalies…

- **Coverage or lack of it**
  - **Missing Attribute**
- Result of omissions while collecting the data.
- A constraint violation if we have null values for attributes where NOT NULL constraint exists.
- Case more complicated where no such constraint exists.
- Have to decide whether the value exists in the real world and has to be deduced here or not.

Coverage Problems

Missing values

Result of omissions while collecting the data. A constraint violation if we have null values for attributes where NOT NULL constraint exists. Case more complicated where no such constraint exists. Have to decide whether the value exists in the real world and has to be deduced here or not.

Missing tuples

Result from omissions of complete entities existent in the mini-world that are not represented by tuples in the data collection. Anomalies could further be classified according to the amount of data accounting for the constraint violation. This can be single values, values within a tuple, values within one or more columns of the data collection or tuples and sets of tuples from different relations.

Why Missing Rows?

- Equipment malfunction (bar code reader, keyboard etc.)
- Inconsistent with other recorded data and thus deleted.
- Data not entered due to misunderstanding/illegibility.
- Data not considered important at the time of entry (e.g. Y2K).

There can be a number of reasons for missing rows or missing data. For example there may be fault with the hardware for recording or inputting the data, this could be a bar code reader which is faulty and missing part of the UPC (Universal Product Code) or on a low tech level, there could be problem with a keyboard (numeric part) that results in entry of different keys as same keys and only one is kept and all others are removed, which infact corresponded to unique keys. In many cases I can not even read what I have written, I am sure I am not alone. If I can't read my own handwriting sometime, than in many instances people will be unable to read my handwriting. This means either the data will not be entered or it will be entered incorrectly, even using automatic means using OCR (Optical Character Reader) soft wares. Another example is the famous (or infamous) Y2K problem, when the first two most significant digits of the year where not recorded, and in many cases it was almost impossible to differentiate between 1901 and 2001!

Handling missing data

- Dropping records.
- “Manually” filling missing values.
- Using a global constant as filler.
- Using the attribute mean (or median) as filler.
- Using the most probable value as filler.

There can be a number of ways of handling missing data, the most convenient being to just drop the records with missing values for certain attributes. The problem with this approach is what do we gain by dropping records with missing values, and how to decide which records to drop i.e. with one missing value or which one or how many missing values? Obviously this is a complex problem, and writing code would not be an easy task, so one easy way out could be to manually fill missing values or use a global constant as a filler. For example if gender is not known for the customers, then filling the gender by (say) NA. For numeric attributes, filling using a global value may make some sense, as it could be the mean or median of the column value. Or this could also be the most probable value to be used as a filler.

Key Based Classification of Problems

- Primary key problems
- Non-Primary key problems

The problems associated with the data can correspond to either the primary keys or non primary keys, as the nature of the problem and the corresponding solution varies with the type of the key involved. In the subsequent slides we will discuss each of them one by one.

Primary key problems

1. Same PK but different data.
2. Same entity with different keys.
3. PK in one system but not in other.
4. Same PK but in different formats.

Some of the problems associated with PK are;

1. Records may have the same primary key but might have different data. This can occur if primary keys are reused or when two organizations or units merge.

2. The same entity might occur with different primary keys. This can easily arise when different segments of an entity design databases independently of one another

3. Data may have a primary key in one system but not in another. The classic example of this kind of error occurs when an entity is represented in more than one source database. It is quite possible that the entity is central to one of the databases and therefore has a primary key field or

fields while the same entity is so peripheral to the purposes of the other database that it is not dignified by being given a primary key.

4. Primary Keys may be intended to be the same but might occur in different formats. Probably, the most widespread instance of this error occurs when NID are used as primary keys: are they character data or numeric; if character data, do they contain dashes?

Non primary key problems…

1. Different encoding in different sources.
2. Multiple ways to represent the same information.
3. Sources might contain invalid data.
4. Two fields with different data but same name.

1. Data may be encoded differently in different sources. The domain of a "gender" field in some database may be {'F', 'M'} or as {"Female", "Male"} or even as {1, 0}.

2. There are often multiple ways to represent the same piece of information. "FAST", "National University", "FAST NU" and "Nat. Univ. of Computers " can all can be found in the literature as representing the same institution.

3. Sources might contain invalid data. A point of sale terminal may require that the sales clerk enters a customer's telephone number. If the customer does not wish to give it, clerks may enter 999-999-9999.

4. Two fields may contain different data but have the same name. There are a couple of ways in which this can happen. "Total Sales" probably means fiscal year sales to one part of an enterprise and calendar year sales to another. The second instance can be much more dangerous. If an application is used by multiple divisions, it is likely that a field that is necessary for one business unit is irrelevant to another and may be left blank by the second unit or, worse, used for otherwise undocumented purposes.

Non primary key problems

- Required fields left blank.
- Data erroneous or incomplete.
- Data contains null values.

5. Required fields may be left blank. Clerical errors account for most of these, but mobile phones did not come into use until early 90's, so contact numbers recorded before then will not have them.

6. Data may be erroneous or inconsistent. The telephone number may be for Lahore but the city listed as Karachi.

7. The data may contain null values. Null values can occur for a wide variety of reasons, the most common of these are:

a. Data that is genuinely missing or unknown,
b. An attribute does not apply to an entity,
c. Data that is pending, or
d. Data that is only partially known.

Automatic Data Cleansing…

1) Statistical
2) Pattern Based
3) Clustering
4) Association Rules

Some of the data cleansing techniques are listed. Let's discuss each of them in detail.

Statistical : Identifying outlier fields and records using the values of mean, standard deviation, range, etc., based on Chebyshev's theorem, considering the confidence intervals for each field. Outlier values for particular fields are identified based on automatically computed statistics. For each field the average and the standard deviation are utilized and based on Chebyshev's theorem those records that have values in a given field outside a number of standard deviations from the mean are identified. The number of standard deviations to be considered is customizable. Confidence intervals are taken into consideration for each field.

Pattern-based: Identify outlier fields and records that do not conform to existing patterns in the data. Combined techniques (partitioning, classification, and clustering) are used to identify patterns that apply to most records. A pattern is defined by a group of records that have similar characteristics ("behavior") for p% of the fields in the data set, where p is a user-defined value (usually above 90).

Clustering : Identify outlier records using clustering based on Euclidian (or other) distance. Existing clustering algorithms provide little support for identifying outliers. However, in some cases clustering the entire record space can reveal outliers that are not identified at the field level inspection. The main drawback of this method is computational time. The clustering algorithms have high computational complexity. For large record spaces and large number of records, the run time of the clustering algorithms is prohibitive.

Association rules: Association rules with high confidence and support define a different kind of pattern. As before, records that do not follow these rules are considered outliers. The power of association rules is that they can deal with data of different types. However, Boolean association rules do not provide enough quantitative and qualitative information.

Lecture-20
Data Duplication Elimination & BSN Method

The *duplicate elimination* task is typically performed after most other transformation and cleaning steps have taken place, especially after having cleaned single-source errors and conflicting representations. It is performed either on two cleaned sources at a time or on a single already integrated data set. Duplicate elimination requires to first identify (i.e. match) similar records concerning the same real world entity. In the second step, similar records are merged into one record containing all relevant attributes without redundancy. Furthermore, redundant records are removed.

Why data duplicated?

A data warehouse is created from heterogeneous sources, with heterogeneous databases (different schema/representation) of the same entity.

The data coming from outside the organization owning the DWH can have even lower quality data i.e. different representation for same entity, transcription or typographical errors.

A data warehouse is created by merging large databases acquired from different sources with heterogeneous representations of information. This raises the issue of data quality, the foremost being identification and elimination of duplicates, crucial for accurate statistical analyses. Other than using own historical/transactional data, it is not uncommon for large businesses to acquire scores of databases each month, with a total size of hundreds of millions to over a billion records that need to be added to the warehouse. The fundamental problem is that the data supplied by various sources typically include identifiers or string data that are either different among different datasets or simply erroneous due to a variety of reasons including typographical or transcription errors or purposeful fraudulent activity, such as aliases in the case of names.

Problems due to data duplication

Data duplication, can result in costly errors, such as:

- False frequency distributions.
- Incorrect aggregates due to double counting.
- Difficulty with catching fabricated identities by credit card companies.

Without accurate identification of duplicated information frequency distributions and various other aggregates will produce false or misleading statistics leading to perhaps untrustworthy new knowledge and bad decisions. Thus this has become an increasingly important and complex problem for many organizations that are in the process of establishing a data warehouse or updating the one already in existence.

Credit card companies routinely assess the financial risk of potential new customers who may purposely hide their true identities and thus their history or manufacture new ones.

The sources of corruption of data are many. To name a few, errors due to data entry mistakes, faulty sensor readings or more malicious activities provide scores of erroneous datasets that propagate errors in each successive generation of data.

Data Duplication: Non-Unique PK

- Duplicate Identification Numbers
- Multiple Customer Numbers

Name	Phone Number	Cust. No.
M. Ismail Siddiqi	*021.666.1244*	*780701*
M. Ismail Siddiqi	*021.666.1244*	*780203*
M. Ismail Siddiqi	*021.666.1244*	*780009*

- Multiple Employee Numbers

Bonus Date	Name	Department	Emp. No.
Jan. 2000	*Khan Muhammad*	*213 (MKT)*	*5353536*
Dec. 2001	*Khan Muhammad*	*567 (SLS)*	*4577833*
Mar. 2002	*Khan Muhammad*	*349 (HR)*	*3457642*

Unable to determine customer relationships (CRM)
Unable to analyze employee benefits trends

What if the customers are divided among sales people to make telephone calls? Maybe the three records of the same person go to three telesales people or maybe to the same one, but arranged as per Cust.No. The point is that no telesales person would know that a single customer is going to get multiple calls from the company, resulting in wastage of time and resources of the company, and at the same time annoying the customer.

In the other case, the employee is same, who has been moving among different departments, and possibly during the process got new employee numbers. He has also been getting bonuses regularly, maybe because every time he appears to be an employee who has never got a bonus. This resulting in loss to the company and an inability of the company to do any meaningful employee benefit analysis.

Data Duplication: House Holding

- Group together all records that belong to the same household.

……	S. Ahad	440, Munir Road, Lahore
……	……………	………………………………
……	Shiekh Ahad	No. 440, Munir Rd, Lhr
……	……………	………………………………
……	Shiekh Ahed	House # 440, Munir Road, Lahore

Why bother?

Why bother?
In a joint family system, many working people live at the same house, but have different bank accounts and ATM cards. If the bank would like to introduce a new service and wants to get in touch with its customers, it would be much better if a single letter is sent to a single household, instead of sending multiple letters. The problem is difficult because multiple persons living at the same house may have written the same address differently even worse, one person with multiple accounts at the same bank may have written his/her name and address differently.

One month is a typical business cycle in certain direct marketing operations. This means that sources of data need to be identified, acquired, conditioned and then correlated or merged within a small portion of a month in order to prepare mailings and response analyses. With tens of thousands of mails to be sent, reducing the numbers of duplicate mails sent to the same household can result in a significant savings.

Data Duplication: Individualization

- Identify multiple records in each household which represent the same individual

………	M. Ahad	440, Munir Road, Lahore
………	……………	………………………………
………	Maj Ahad	440, Munir Road, Lahore

Address field is standardized, is this by coincidence? This could be your luck data, but don't count on it i.e. this is very unlikely to be the default case.

Formal definition & Nomenclature

- Problem statement:
  - "Given two databases, identify the potentially matched records Efficiently and Effectively"
- Many names, such as:
  - Record linkage
  - Merge/purge
  - Entity reconciliation
  - List washing and data cleansing.
- Current market and tools heavily centered towards customer lists.

Within the data warehousing field, data cleansing is applied especially when several databases are merged. Records referring to the same entity are represented in different formats in the different data sets or are represented erroneously. Thus, duplicate records will appear in the merged database. The issue is to identify and eliminate these duplicates. The problem is known as the *merge/purge problem.* Instances of this problem appearing in literature are called record linkage, semantic integration, instance identification, or object identity problem.

Data cleansing is much more than simply updating a record with good data. Serious data cleansing involves decomposing and reassembling the data. One can break down the cleansing into six steps: elementizing, standardizing, verifying, matching, house holding, and documenting. Although data cleansing can take many forms, the current marketplace and the current technology for data cleansing are heavily focused on customer lists.

Need & Tool Support

- Logical solution to dirty data is to clean it in some way.
- Doing it manually is very slow and prone to errors.
- Tools are required to do it "cost" effectively to achieve reasonable quality.
- Tools are there, some for specific fields, others for specific cleaning phase.
- Since application specific, so work very well, but need support from other tools for broad spectrum of cleaning problems.

For existing data sets, the logical solution to the dirty data problem is to attempt to cleanse the data in some way. That is, explore the data set for possible problems and endeavor to correct the errors. Of course, for any real world data set, doing this task "by hand" is completely out of the question given the amount of man hours involved. Some organizations spend millions of dollars per year to detect data errors. A manual process of data cleansing is also laborious, time consuming, and itself prone to errors. The need for useful and powerful tools that automate or greatly assist in the data cleansing process are necessary and may be the only practical and cost effective way to achieve a reasonable quality level in an existing data set.

A large variety of tools is available in the market to support data transformation and data cleaning tasks, in particular for data warehousing. Some tools concentrate on a specific domain, such as cleaning name and address data, or a specific cleaning phase, such as data analysis or duplicate elimination. Due to their restricted domain, specialized tools typically perform very well but must be complemented by other tools to address the broad spectrum of transformation and cleaning problems.

Overview of the Basic Concept

- In its simplest form, there is an identifying attribute (or combination) per record for identification.
- Records can be from single source or multiple sources sharing same PK or other common unique attributes.
- Sorting performed on identifying attributes and neighboring records checked.
- What if no common attributes or dirty data?
- The degree of similarity measured numerically, different attributes may contribute differently.

In the simplest case, there is an identifying attribute or attribute combination per record that can be used for matching records, e.g., if different sources share the same primary key or if there are other common unique attributes. For example people in income tax department get customer details from the phone company and the Electricity Company and want to identify customers who have high electricity and high telephone bill but do not pay tax accordingly. Would be the common field, NID, they have changed over time, how about addresses, too many ways to write addresses so have to figure out common attributes.

Instance matching between different sources is then achieved by a standard equi-join on the identifying attribute(s), if you are very, very lucky. In the case of a single data set, matches can be determined by sorting on the identifying attribute and checking if neighboring records match. In both cases, efficient implementations can be achieved even for large data sets. Unfortunately, without common key attributes or in the presence of dirty data such straightforward approaches are often too restrictive. For example, consider a rule that states person records are likely to correspond if name and portions of the address match.

The degree of similarity between two records, often measured by a numerical value between 0 and 1, usually depends on application characteristics. For instance, different attributes in a matching rule may contribute different weight to the overall degree of similarity.

Basic Sorted Neighborhood (BSN) Method

- Concatenate data into one sequential list of N records
- **Steps 1:** Create Keys
    - Compute a key for each record in the list by extracting relevant fields or portions of fields
    - Effectiveness of the this method highly depends on a properly chosen key
- **Step 2:** Sort Data
    - Sort the records in the data list using the key of step 1
- **Step 3:** Merge
    - Move a fixed size window through the sequential list of records limiting the comparisons for matching records to those records in the window
    - If the size of the window is *w* records then every new record entering the window is compared with the previous *w-1* records.

1. **Create Keys:** Compute a key for each record in the list by extracting relevant fields or portions of fields. The choice of the key depends upon an error model that may be viewed as knowledge intensive and domain specific for the effectiveness of the sorted neighborhood method. This highly depends on a properly chosen key with the assumption that common but erroneous data will have closely matching keys.

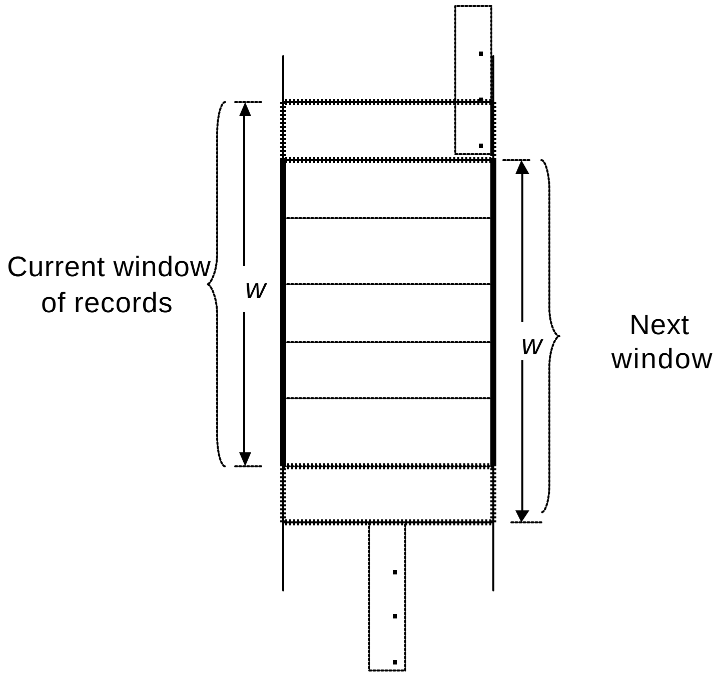
2. **Sort Data**: Sort the records in the data list using the key of step-1.

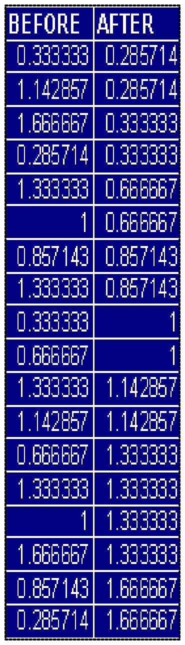
3. **Merge:** Move a fixed size window through the sequential list of records limiting the comparisons for matching records to those records in the window. If the size of the window is

w records as shown in Figure 20.1, then every new record entering the window is compared with the previous *w - 1* records to find “matching” records. The first record in the window slides out of the window.

Note that in the absence of a window, the comparison would cover all the sorted records i.e. a pair-wise comparison, resulting in O (n^2) comparisons.

BSN Method : Sliding Window





BEFORE	AFTER
0.333333	0.285714
1.142857	0.285714
1.666667	0.333333
0.285714	0.333333
1.333333	0.666667
1	0.666667
0.857143	0.857143
1.333333	0.857143
0.333333	1
0.666667	1
1.333333	1.142857
1.142857	1.142857
0.666667	1.333333
1.333333	1.333333
1	1.333333
1.666667	1.333333
0.857143	1.666667
0.285714	1.666667

Figure-20.1: BSN Method: Sliding Window

Figure-20.1 shows the outcome of the sliding window i.e. the IDs sorted after completion of the run. The outcome will also depend on the window size, as it may so happen that when the window size is relatively small and the keys are dispersed then some (or most o f the keys) may not land in their corresponding neighborhood, thus defeating the purpose of the BSN method. One way of overcoming this problem is working with different window sizes in a multipass approach.

Complexity Analysis of BSN Method

- Time Complexi ty: O(n log n)
  - O (n) for Key Creation
  - O (n log n) for Sorting
  - O (w n) for matching, where $w \leq 2 \leq n$
  - Constants vary a lot
- At least three passes required on the dataset.

- For large sets disk I/O is detrimental.
- Complexity or rule and window size detrimental.

When this procedure is executed serially as a main memory based process the create keys phase is an O(n) operation the sorting phase is O(n log n) and the merging phase is O(wn) where n is the number of records in the database. Thus, the total time complexity of this method is O (n log n) if w < $\lceil \log n \rceil$, O(wn) otherwise. However the constants in the equations differ greatly. It could be relatively expensive to extract relevant key values from a record during the create key phase. Sorting requires a few machine instructions to compare the keys. The merge phase requires the application of a potentially large number of rules to compare two records and thus has the potential for the largest constant factor.

Notice that w is a parameter of the window scanning procedure. The legitimate values of w may range from 2 (whereby only two consecutive elements are compared) to n (whereby each element is compared to all others). The latter case pertains to the full quadratic O (n^2) time process at the maximal potential accuracy. The former case (where *w* may be viewed as a small constant relative to *n)* pertains to optimal time performance (only O (n) time) but at minimal accuracy.

Note however that for very large databases the dominant cost is likely disk I/O and hence the number of passes over the data set. In this case at least three passes would be needed one pass for conditioning the data and preparing keys at least a second pass likely more for a high speed sort, and a final pass for window processing and application of the rule program for each record entering the sliding window. Depending upon the complexity of the rule program and window size w the last pass may indeed be the dominant cost.

BSN Method: Selection of Keys

- Selection of Keys
  - Effectiveness highly dependent on the key selected to sort the records middle name vs. last name,
  - A key is a sequence of a subset of attributes or sub-strings within the attributes chosen from the record.
  - The keys are used for sorting the entire dataset with the intention that matched candidates will appear close to each other.

First	**Middle**	**Address**	**NID**	**Key**
Muhammed	Ahmad	440 Munir Road	34535322	AHM440MUN345
Muhammad	Ahmad	440 Munir Road	34535322	AHM440MUN345
Muhammed	Ahmed	440 Munir Road	34535322	AHM440MUN345
Muhammad	Ahmar	440 Munawar Road	34535334	AHM440MUN345

Table-20.1: BSN Method: Selection of Keys

The key consists of the concatenation of several ordered fields or attributes in the data. The first three letters of a middle name are concatenated with the first three letters of the first name field followed by the address number field and all of the consonants of the road name. This is followed by the first three digits of the National ID field as shown in Table 20.1.

These choices are made since the key designer determined that family names are not very discriminating (Khan, Malik andCheema). The first names are also not very discriminating and are typically misspelled (Muhammed, Muhammad, Mohamed) due to mistakes in vocalized sounds vowels but middle names are typically more uncommon and less prone to being misunderstood and hence less likely to be recorded incorrectly.

The keys are now used for sorting the entire dataset with the intention that all equivalent or matching data will appear close to each other in the final sorted list. Notice in table 20.1, how the first and second records are exact duplicates while the third is likely to be the same person but with a misspelled middle name i.e. Ahm**e**d instead of Ahm**a**d. We would expect that this phonetically based mistake will be caught by a reasonable equational theory. However the fourth record although having the exact same key as the prior three records appears unlikely to be the same person.

BSN Method: Problem with keys

- Since data is dirty, so keys WILL also be dirty, and matching records will not come together.
- Data becomes dirty due to data entry errors or use of abbreviations. Some real examples are as follows:

  Technology

  Tech.

  Techno.

  Tchnlgy
- Solution is to use external standard source files to validate the data and resolve any data conflicts.

Since the data is dirty and the keys are extracted directly from the data then the keys for sorting will also be dirty. Therefore the process of sorting the records to bring matching records together will not be as effective. A substantial number of matching records may not be detected in the subsequent window scan.

Data can become dirty due to data entry errors. To speed-up data entry abbreviations are also used (all taken from the telephone directory.), such as:

Tehcnology, Tech. Tehcno. Tchnlgy

The above is a typical case of a non-PK error of same entity with multiple representations. To ensure the correctness of data in the database, solution is using external source files to validate the data and resolve any data conflicts.

BSN Method: Problem with keys (e.g.)

If contents of fields are not properly ordered, similar records will NOT fall in the same window.
Example: Records 1 and 2 are similar but will occur far apart.

No	Name	Address	Gender
1	N. Jaffri, Syed	No. 420, Street 15, Chaklala 4, Rawalpindi	M
2	S. Noman	420, Scheme 4, Rwp	M
3	Saiam Noor	Flat 5, Afshan Colony, Saidpur Road, Lahore	F

Solution is to TOKENize the fields i.e. break them further. Use the tokens in different fields for sorting to fix the error.
Example: Either using the name or the address field records 1 and 2 will fall close.

No	Name	Address	Gender
1	Syed N Jaffri	420 15 4 Chaklala No Rawalpindi Street	M
2	Syed Noman	420 4 Rwp Scheme	M
3	Saiam Noor	5 Afshan Colony Flat Lahore Road Saidpur	F

We observe that characters in a string can be grouped into meaningful pieces. We can often identify important components or tokens within a Name or Address field by using a set of delimiters such as space and punctuations. Hence we can first tokenize these fields and then sort the tokens within these fields. Records are then sorted on the select key fields; note that this approach is an improvement over the standard BSN method.

BSN Method: Matching Candidates

Mergin g of records is a complex inferential process.

Example-1: Two persons with names spelled nearly but not identically, have the exact same address. We infer they are same person i.e. Noma Abdullah and Noman Abdullah.

Example-2: Two persons have same National ID numbers but names and addresses are completely different. We infer same person who changed his name and moved or the records represent different persons and NID is incorrect for one of them.

Use of further information such as age, gender etc. can alter the decision.

Example-3: Noma-F and Noman-M we could perhaps infer that Noma and Noman are siblings i.e. brothers and sisters. Noma-30 and Noman-5 i.e. mother and son.

The comparison of records during the merge phase to determine their equivalence is a complex inferential process that considers and requires much more information in the compared records than the keys used for sorting. For example suppose two person names are spelled nearly but not identically, and have the exact same address. We might infer they are the same person. For example Noma Abdullah and Noman Abdullah could be the same person if they have the same address.

On the other hand suppose two records have exactly the same National ID numbers but the names and addresses are completely different. We could either assume the records represent the same person who changed his name and moved or the records represent different persons and the NID number field is incorrect for one of them. Without any further information we may perhaps assume the latter. The more information there is in the records the better inferences can be made. For example, if gender and age information is available in some field of the data (Noma-F, Noman-M) we could perhaps infer that Noma and Noman are siblings.

BSN Method: Equational Theory

To specify the inferences we need equational Theory.

- Logic is NOT based on string equivalence.
- Logic based on domain equivalence.
- Requires declarative rule language.

To specify the inferences as discussed above we need an equational theory in which the logic is based not on the string equivalence but on the domain equivalence. A natural approach to specifying an equational theory and making it practical too would be the use of a declarative rule language. Rule languages have been effectively used in a wide range of applications requiring inference over large data sets. Much research has been conducted to provide efficient means for their compilation and evaluation and this technology can be exploited here for purposes of data cleansing efficiently.

BSN Method: Rule Language

- Rule Language Example

```
Given two records r1 and r2
IF the family_name of r1 equals the family_name of r2
AND the first names differ slightly
AND the address of r1 equals the address of r2
THEN r1 is equivalent to r2
```

This is rather self explanatory. The rule merely states that if for two records the last names and the addresses are same, but the first names differ slightly, then both the records are same. This makes sense, as using default values last names can be easily fixed, and sorting the address and then using default values can fix most part of the address too, as the addresses are repeating.

BSN Method: Mis-Matching Candidates

- Transposition of names
- How do you specify "Differ Slightly"?
  - Calculate on the basis of a distance function applied to the family_name fields of two records
  - Multiple options for distance functions

Notice that rules do not necessarily need to compare values from the same attribute or same domain. For instance to detect a transposition in a persons name we could write a rule that compares the first name of one record with the last name of the second record and the last name of the first record with the first name of the second record.

BSN Method: Distance Metrics

- Distance Functions
  - Phonetic distance
  - Typewriter distance
  - Edit Distance
- A widely used metric to define string similarity
  - Ed(s1,s2)= minimum # of operations (insertion, deletion, substitution) to change s1 to s2
- Example:

  s1: Muhammad Ehasn
  s2: Muhammed Ahasn
  ed(s1,s2) = 2

The implementation is based upon the computation of a distance function applied to the first name fields of two records and the comparison of its results to a threshold to capture obvious typographical errors that may occur in the data. The selection of a distance function and a proper threshold is also a knowledge intensive activity that demands experimental evaluation. An improperly chosen threshold will lead to either an increase in the number of falsely matched records or to a decrease in the number of matching records that should be merged. A number of alternative distance functions for typographical mistakes are possible, including distances based upon (i) edit distance (ii) phonetic distance and (iii) typewriter distance.

Limitations of BSN Method

- BSN Method Limitations
    - No single key is sufficient to catch all matching records.
    - Fields that appear first in the key have higher discriminating power than those appearing after them.
    - If NID number is first attribute in the key 81684854432 and 18684854432 are highly likely to fall in windows that are far apart.
- Two Possible Modifications to BSN Method
    - Increase *w*, the size of window
    - Multiple passes over the data set

In general no single key will be sufficient to catch all matching records. The attributes or fields that appear first in the key have higher discriminating power than those appearing after them. Hence if the error in a record occurs in the particular field or portion of the field that is the most important part of the key there may be little chance a record will end up close to a matching record after sorting. For instance if an employee has two records in the database one with NID number 81684854432 and another with NID number 18684854432 (the first two numbers were transposed) and if the NID number is used as the principal field of the key then it is very unlikely both records will fall under the same window i.e. the two records with transposed NID numbers will be far apart in the sorted list and hence they may not be merged. As it was empirically established that the number of matching records missed by one run of the sorted neighborhood method can be large unless the neighborhood grows very large.

Increasing w

- Increases the computational complexity.
- Quality does not increase dramatically.
- Not useful unless the window spans the entire database.
    - W spanning the entire set of rows is infeasible under strict time and cost constraints.
    - What would be the computational complexity?
    - O(w2)

Increasing *w* clearly this increases the computational complexity and as discussed in the next section does not increase dramatically the number of similar records merged in the test cases we ran unless of course the window spans the entire database which we have presumed is infeasible under strict time and cost constraints.

Multi-pass Approach

- Several independent runs of the BSN method each time with a different key and a relatively small window.
- Each independent run will produce a set of pairs of records which can be merged (takes care of transposition errors)
- Apply the transitive closure property to those pairs of records.
    - If records a and b are found to be similar and at the same time records b and c are also found to be similar the transitive closure step can mark a and c to be similar.
- The results will be a union of all pairs discovered by all independent runs with no duplicates plus all those pairs that can be inferred by transitivity of equality.

The alternative strategy we implemented is to execute several independent runs of the sorted neighborhood method each time using a different key and a relatively small window. We call this strategy the multipass approach. For instance in one run we use the address as the principal part of the key while in another run we use the last name of the employee as the principal part of the key. Each independent run will produce a set of pairs of records which can be merged. We then apply the transitive closure to those pairs of records.

The results will be a union of all pairs discovered by all independent runs with no duplicates plus all those pairs that can be inferred by transitivity of equality. The reason this approach works for the test cases explored here has much to do with the nature of the errors in the data. Transposing the first two digits of the NID number leads to nonmergeable records as we noted. However in such records the variability or error appearing in another field of the records may indeed not be so large. Therefore although the NID numbers in two records are grossly in error the name fields may not be. Hence first sorting on the name fields as the primary key will bring these two records closer together, lessening the negative effects of a gross error in the social security field.

Lecture-21
Introduction to Data Quality Management (DQM)

Data is the "fuel" on which a Data Warehouse (DWH) runs. One of the main reasons for the failure of DWH deployments is data quality or lack of it. Unfortunately the problems of data quality are usually underestimated, leading to bad decisions, traumatic results and expensive fixes.

There can be a whole course on data quality. The goal of this lecture is to expose the reader to the data quality issues, and to motivate them to understand more about quality management and quality control. This will enable the reader to relate quality management to the DWH, because usually quality control relates to manufacturing than Data Warehousing.

Consider the analogy of a car. You may have the best car, the best road conditions and the best driver on the seat. But if the fuel is contaminated, or is not of the right octane, either the car will not run, or will run, but not give the right performance.

What is Quality? Informally

Some things are better than others i.e. they are of higher quality. How much "better" is better?

Is the right item the best item to purchase? How about after the purchase?

What is quality of service? The bank example

The best way to look at data quality is to look at what quality means in the general marketplace and then translate what quality means for data. As consumers, human beings consciously or subconsciously judge the "quality" of things during their life-time. A conscious application of quality measurement is when a person compares products in a store and chooses one of them as the "right" product. "Right" here means selecting the product that best meets his/her overall needs. Note that the product bought may not necessarily have the best features in every category. After purchase, people establish quality based on a product-price comparison i.e. whether that product for its price met their expectations as per its intended use?

What is Quality? Formally

Quality is conformance to requirements"

P. Crosby, "Quality is Free" 1979

Degree of excellence"

Webster's Third New International Dictionary

Conformance to requirements does not imply simply conformation to written specifications. Customers' requirements may be formal and written, or informal mental expectations of meeting their purpose or satisfying their needs.

If a product meets formally defined "requirement specifications," yet fails to be a quality product from the customers' perspective, this means the requirements were defective.

If an application is designed and built to *meet* the functional requirements signed off by the business sponsors, and during formal testing the business sponsors reject the application as not meeting their needs, what does that mean? Either the requirements specification or the analysis and design process is defective.

What is Quality? Examples from Auto Industry

Quality means meeting customer's needs, not necessarily exceeding them.

Quality means improving things customers *care about*, because that makes their lives easier and more comfortable.

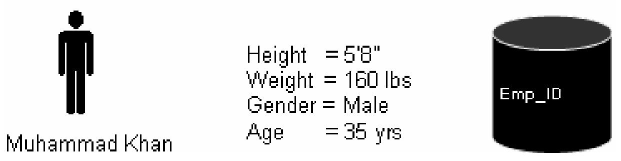
Why example from auto-industry?

The luxury automobile producer Rolls Royce went bankrupt in the early 1980s. Analysis revealed that, among other things, Rolls Royce was improving components that the luxury automobile customers felt were irrelevant and polishing parts they did not care about. This drove the price beyond what the luxury automobile customer felt was value for their money.

On the other hand, when Lexus decided to make its first major redesign of its highly rated L8 400 luxury automobile, company representatives asked for help from their existing customers. They even visited the homes of a variety of L8 400 owners to observe home furnishings, what kind of leather they had on their brief cases, and other minute details to get an idea of their customer's subconscious expectations.

What is Data Quality?

What is Data?



All data is an abstraction of something real

Intrinsic Data Quality
Electronic reproduction of reality.

Realistic Data Quality
Degree of utility or value of data to business

Figure-21.1: What is Data Quality?

What is data?
Understand that all data is an abstraction or a representation of something real. Data is an equivalent reproduction of something real. Figure 21.1 shows an employee data. Data in a

database or data warehouse has no actual value; it only has potential value. Data has a real value only when someone uses it to do something useful; for example, to send the utility bill to customer at the correct address for timely payment, or to successfully go with the change in a product based on the knowledge of customer likes/dislikes.

What Is Data Quality?

There are two significant definitions of data quality. One is its intrinsic quality, and the other is its realistic quality. Intrinsic data quality is the correctness or accuracy of data. Realistic data quality is the value that correct data has in supporting the work of the business or enterprise. Data that does not help or enable the enterprise to accomplish its mission has no quality, no matter how accurate it is.

Intrinsic Data Quality

Intrinsic data quality, simply stated, is data accuracy. Intrinsic data quality is the degree to which data accurately reflects the real-world object that the data represents. If all facts that an organization needs to know about an entity are accurate, that data has intrinsic quality-it is an electronic reproduction of reality. Intrinsic data quality means that data is correct.

Realistic Data Quality

Realistic data quality is the degree of utility and value the data has to support the enterprise processes that enable accomplishing enterprise objectives. Fundamentally, realistic data quality is the degree of customer satisfaction generated by the knowledge workers who use it to do their jobs. Realistic data quality is the degree to which data enables knowledge workers to meet enterprise objectives efficiently and effectively.

Data Quality & Organizations

Intelligent Learning Organization:
High-quality information is an *open, shared* resource with *value-adding* processes.

The dysfunctional learning organization: Low-quality data is a *proprietary* resource with *cost-adding* processes.

The intelligent learning organization is one that maximizes both its experience and its information resources in the learning process. The intelligent learning organization shares information openly across the enterprise in a way that maximizes the throughput of the entire organization. There are no procedural walls, and the data users capture data and retrieve makes the business to grow.

In the Information Age, the dysfunctional learning organization is at a distinct disadvantage. The term dysfunctional means "impaired or abnormal functioning."

Orr's Laws of Data Quality

Law #1 - "Data that is not used cannot be correct!"

Law #2 - "Data quality is a function of its use, not its collection!"

Law #3 - "Data will be no better than its most stringent use!"

Law #4 - "Data quality problems increase with the age of the system!"

Law #5 – "The less likely something is to occur, the more traumatic it will be when it happens!"

We will only know about the quality of the data (good or bad) once we start using it. IF the data has never been used, then it would be naïve and self deceiving to assume that it is of the right quality. A realistic approach would be to assume that it is not of the right quality, IF it ha snot been used.

As the data is continued to be used, its quality issues come up and as a consequence its quality gets improved. A good analogy could be that of a machine, a well oiled machine which is being used is likely to continue to work as compared to a large machine that has not been used for a long time.

The third law kind of supports or goes with the second law i.e. the more stringent use of the data is, the more likely that the quality issues that crop up will be identified and fixed. Consider the analogy of a map. A street map given in an atlas is fine as long as using it you can find the major shopping areas, however, you may be surprised that you may not be able to find your house using that map, because the map was not meant to handle the stringent use of locating your house.

Law-4 can have two possible exp lanations. One is rather obvious i.e. the shelf life of storage medium which is very low for diskettes, comparatively better for magnetic tapes and very high for CDs. Depending on which medium was used to store the data will dictate the data quality. The other possibility is the meta data or people who are aware of the data anomalies but leave the organization over a period of time by retiring or changing jobs or die.

Total Quality Management (TQM)

Philosophy of involving all for systematic and continuous improvement.

It is customer oriented. Why?

TQM incorporates the concept of product quality, process control, quality assurance, and quality improvement.

Quality assurance is NOT Quality improvement

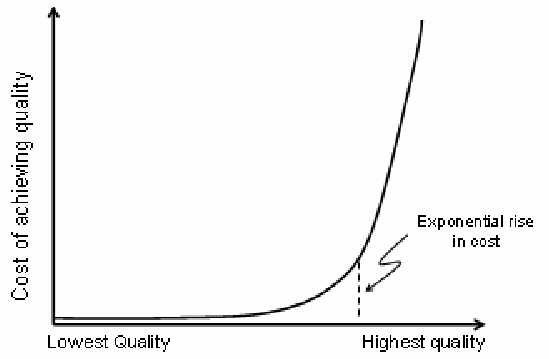
TQM approach is advocating the involvement of all employees in the continuous improvement process, the ultimate goal being the customer satisfaction.

The TQM philosophy of management is customer-oriented. All members of a total quality management (control) organization strive to systematically manage the improvement of the organization through the ongoing participation of all employees in problem solving efforts across functional and hierarchical boundaries.

Quality assurance is a system of activities that assures conformance of product to pre-established requirements.

Quality improvement is making all efforts to increase effectiveness and efficiency in meeting accepted customer expectations.

Co$t of fixing data quality



- Defect minimization is economical.
- Defect elimination is very very expensive

Figure-21.2: Co$t of fixing data quality

One can ask for the stars i.e. perfect data quality, only if there is no associated cost, in such a case this may be a perfectly legitimate requirement. However, in real world, such a requirement is NOT realistic. Why? If you give me an infinite amount of money, I will personally look at each and every record and data element. But this is not realistic; you don't have enough money to pay for that. Even if you had the money, it will take an inordinate amount of time to check each and every record. By the time I am finished checking/fixing data, trends may have changed, and the question that needed the data may no longer be interesting any more.

Note that as the data quality is pushed beyond a certain limit, law of diminishing returns sets in i.e. increasing the input does not results in corresponding increase in output, actually the output starts to go down in this the cost-quality ratio, as shown in Figure 21.2.

Co$t of Data Quality Defects

- Controllable Costs
  - Recurring costs for analyzing, correcting, and preventing data errors
- Resultant Costs
  - Internal and external failure costs of business opportunities missed.
- Equipment & Training Costs

The costs are immense. Inaccurate data will lead to bad decisions, and bad decisions can never be profitable. Unless the fingers of the CEO are on the pulse of the business, that business can not prosper. The pulse of the business is access to reliable data, without which a business can not be managed.

Direct costs of data quality to a data intensive organization include the following:

1. Controllable costs: If possible, compare two or more alternatives for improving data quality. Estimate the controllable, equipment, and training costs for each alternative. Include an estimate of labor hours devoted to prevent, appraise, and correct problems.

2. Resultant costs: Missing business opportunities. Mis-communicating within the business or with end customers and other information stakeholders. Sometimes resultant costs and indirect costs are more difficult to quantify. Assess these costs wherever possible to adequately measure the impacts of poor data quality. For example, the inability to match payroll records to the official employment records can cost millions in payroll overpayments to retirees, personnel on leave without pay status, and "ghost" employees. Inability to correlate purchase orders to invoices may be a major problem in unmatched disbursements. Resultant costs, such as payroll overpayments and unmatched disbursements, may be significant enough to warrant extensive changes in processes, systems, policy and procedure, and information system data designs.

3. Equipment and training costs: Costs for data quality tools, ancillary hardware and software, and training required to prevent, appraise, and correct data quality problems.

Where data quality is critical?

- **Almost everywhere, some examples:**
- Marketing communications.
- Customer matching.
- Retail house-holding.
- Combining MIS systems after acquisition.

One of the reasons for the lack of planning to address data quality issues is that operational system people don't sufficiently consider the business impact of bad data. Basically their goal is to increase the number of (say) claims processed per hour. To put it more positively, they must better appreciate the opportunities of DWH applications that are critically dependent on data quality. Following is a sampler of some CRM-based applications.

Marketing Communications: If you want to understand who your customers are, and if you want to communicate effectively with them on the phone, mail and via email, you must have an extraordinarily clean customer list. You destroy your credibility by using absurd or misspelled addresses or names or by sending multiple letters to the same person. Even worse, if the address is invalid in some way, the letter never reaches the intending recipient.

Customer Matching: You want to find the same customer when the customer buys a second or a third product from you. Customer matching is a major issue in banking and healthcare, where separate customer (or patient) encounters are often listed separately. The average bank has great difficulty listing all of the separate accounts of a given individual, although the reverse process of listing all the individuals in a specific account is pretty straightforward.

Retail House-holding: You want to find a group of people who comprise a family, each of whom is your customer. When you correctly identify the household, you can communicate coherently with its members. At that time you can identify the overall needs of the household and suggest an effective consolidation or expansion of products. This process is called "cross-selling." Cross-selling your own customer base is recognized as one of the most effective ways to increase sales.

Combining Information Systems after an Acquisition: An increasingly common MIS problem is merging customer lists and product lists from incompatible information systems after an acquisition. Organizationally, it may be easy to imagine combining staffs and moving everything to the corporate system after an acquisition, but what do you do with the data itself? Sometimes the customer list is the most valuable asset of the acquired organization.

Characteristics or Dimensions of Data Quality

Data Quality Characteristic	**Definition**
Accuracy	Qualitatively assessing lack of error, high accuracy corresponding to small error.
Completeness	The degree to which values are present in the attributes that require them.
Consistency	A measure of the degree to which a set of data satisfies a set of constraints.
Timeliness	A measure of how current or up to date the data is.
Uniqueness	The state of being only one of its kind or being without an equal or parallel.
Interpretability	The extent to which data is in appropriate languages, symbols, and units, and the definitions are clear.
Accessibility	The extent to which data is available, or easily and quickly retrievable
Objectivity	The extent to which data is unbiased, unprejudiced, and impartial

95% accurate and 100% complete
OR
100% accurate and 95% complete

Which is better?

Depends on data quality (i) tolerances,
the (ii) corresponding application and the (iii) cost of achieving that data quality vs. the (iv) business value.

Lecture-22
DQM: Quantifying Data Quality

Background

How good is a company’s data quality? Answering this question requires usable data quality metrics. Studies have confirmed data quality is a multi-dimensional concept. Companies must deal with both the subjective perceptions of the individuals involved with the data, and the objective measurements based on the data set in question.

Subjective data quality assessments reflect the needs and experiences of stakeholders: the collectors, custodians, and consumers of data products. This was the approach adopted for assuring the quality of products too.

More on Characteristics of Data Quality

Data Quality Dimensions
Believability
Appropriate Amount of Data
Timeliness
Accessibility
Objectivity
Interpretability
Uniqueness

Data Quality Assessment Techniques

- Ratios
- Min-Max

When performing objective assessments, companies follow a set of principles to develop metrics specific to their needs, there is hard to have “one size fits all” approach. Three pervasive functional forms are (i) simple ratio, (ii) min or max operation, and (iii) weighted average.

Refinements of these functional forms, such as addition of sensitivity parameters, can be easily incorporated. Often, the most difficult task is precisely defining a dimension, or the aspect of a dimension that relates to the company's specific application. Formulating the metric is straightforward once this task is complete.

Data Quality Assessment Techniques

- **Simple Ratios**
  - Free-of-error
  - Completeness
  - Consistency

Simple Ratios

The simple ratio measures the ratio of desired outcomes to total outcomes. Since most people measure exceptions, however, a preferred form is the number of undesirable outcomes divided by total outcomes subtracted from 1. This simple ratio adheres to the convention that 1 represents the most desirable and 0 the least desirable score. Although a ratio illustrating undesirable outcomes gives the same information as one illustrating desirable outcomes, but experience suggests managers prefer the ratio showing positive outcomes, since this form is useful for longitudinal comparisons illustrating trends of continuous improvement. Many traditional data quality metrics, such as *free-of-error*, *completeness*, and *consistency* take this form. Other dimensions that can be evaluated using this form include concise representation, relevancy, and ease of manipulation.

The free-of-error dimension represents data correctness. If one is counting the data units in error, the metric is defined as the number of data units in error divided by the total number of data units subtracted from 1. In practice, determining what constitutes a data unit and what is an error requires a set of clearly defined criteria. For example, the degree of precision must be specified. It is possible for an incorrect character in a text string to be tolerable in one circumstance but not in another.

The completeness dimension can be viewed from many perspectives, leading to different metrics. At the most abstract level, one can define the concept of schema completeness, which is the degree to which entities and attributes are not missing from the schema. At the data level, one can define column completeness as a function of the missing values in a column of a table. A third type is called population completeness. If a column should contain at least one occurrence of all 34 districts of Punjab, for example, but it only contains 30 districts, then we have population incompleteness. Each of the three types (schema completeness, column completeness, and population completeness) can be measured by taking the ratio of the number of incomplete items to the total number of items and subtracting from 1.

The consistency dimension can also be viewed from a number of perspectives, one being consistency of the same (redundant) data values across tables. Codd's referential Integrity constraint is an instantiation of this type of consistency. As with the previously discussed dimensions, a metric measuring consistency is the ratio of violations of a specific consistency type to the total number of consistency checks subtracted from one.

Data Quality Assessment Techniques

- **Min-Max**
  - Believability
  - Appropriate Amount of Data

Min or Max Operation

To handle dimensions that require the aggregation of multiple data quality indicators (variables), the minimum or maximum operation can be applied. One computes the minimum (or maximum) value from among the normalized values of the individual data quality indicators. The min operator is conservative in that it assigns to the dimension an aggregate value no higher than the value of its weakest data quality indicator (evaluated and normalized to between 0 and 1). The maximum operation is used if a liberal interpretation is warranted. The individual variables may be measured using a simple ratio. Two interesting examples of dimensions that can make use of the min operator are believability and appropriate amount of data. The max operator proves useful in more complex metrics applicable to the dimensions of timeliness and accessibility.

Believability is the extent to which data is regarded as true and credible. Among other factors, it may reflect an individual's assessment of the credibility of the data source, comparison to a commonly accepted standard, and previous experience. Each of these variables is rated on a scale from 0 to 1, and overall believability is then assigned as the minimum value of the three. Assume the believability of the data source is rated as 0.6; believability against a common standard is 0.8; and believability based on experience is 0.7. The overall believability rating is then 0.6 (the lowest number). As indicated earlier, this is a conservative assessment. An alternative is to compute the believability as a weighted average of the individual components.

A working definition of the *appropriate amount of data* should reflect the data quantity being neither too little nor too much. A general metric that embeds this tradeoff is the minimum of two simple ratios: the ratio of the number of data units provided to the number of data units needed, and the ratio of the number of data units needed to the number of data units provided.

Data Quality Assessment Techniques

- **Min-Max**
  - Timeliness
  - Accessibility

Timeliness reflects how up-to-date the data is with respect to the task it's used for. A general metric to measure timeliness is to measure the maximum of one of two terms: 0 and one minus the ratio of currency to volatility i.e. Max(0, 1-C/V) . Here, currency (C) is defined as the age (A) plus the delivery time (Dt) minus the input time (It) C = A + Dt - It. Volatility refers to the length of time data remains valid; delivery time refers to when data is delivered to the user; input time refers to when data is received by the system, and age refers to the age of the data when first received by the system.

A similarly constructed metric can be used to measure *accessibility*, a dimension reflecting ease of data attainability. The metric emphasizes the time aspect of accessibility and is defined as the maximum value of two terms: 0 or one minus the time interval from request by user to delivery to user divided by the time interval from request by user to the point at which data is no longer useful. Again, a sensitivity factor in the form of an exponent can be included.

Data Quality Validation Techniques

- Referential Integrity (RI).
- Attribute domain.
- Using Data Quality Rules.
- Data Histograming.

Some of the data validation techniques have been listed. We will discuss each of the technique in detail.

Referential Integrity Validation

Example:

How many outstanding payments in the DWH without a corresponding customer_ID in the customer table?

While doing total quality measurement, you measure RI every week (or month) and hopefully the number of orphan records will be going down, as you will be fine tuning the processes to get rid of the RI problems. Remember, RI problem is peculiar to a DWH, this will not happen in a traditional OLTP system.

Business Case for RI

Not very interesting to know number of outstanding payments from a business point of view.

Interesting to know the actual amount outstanding, on per year basis, per region basis…

It is important to measure the actual (i.e. business) impact of the data quality problem. Knowing how many claims are orphan might be interesting from an analysis point of view. But knowing how many dollars are associated with orphan claims has a business value. If there are too many orphan claims, but too many dollars are not associated with those claims, then it does not have a business impact. We are always trying to relate with the business impact.

Performance Case for RI

Cost of enforcing RI is very high for large volume DWH implementations, therefore:

- Should RI constraints be turned OFF in a data warehouse? or
- Should those records be "discarded" that violate one or more RI constraints?

Cost of transactional RI enforcement is very high for large volume DWH implementations. Assume a company with a multi million row customer table i.e. *n* rows. Checking for RI using a naive approach would take *O(n)* time, using a smart technique with some kind of a tree data structure would require *O(log n)* time, ignoring RI altogether will take *O(1)* time. Therefore, for a chain store with several million transactions per day, every time spending *O(log n)* time will turn out to be extremely expensive.

Another point worth noting is, are you willing to "throw away" rows that violate one or more constraints? May be not, because this will result in losing more information without gaining anything in return.

The bottom line is, most DWH implementations today do not use RI constraints enforced by the database, but as TQM methods improve overall data quality and database optimizers become more sophisticated in the use of constraints, this will become a more attractive option.

3 steps of Attribute Domain Validation

The occurrences of each domain value within each coded attribute of the database.

Actual content of attributes against set of valid values.

Exceptions to determine cause and impact of the data quality defects.

Step-1: This will be done for every single table. Run a script by table, by column and collect all the values in that column and do a count on them, so that you know for each domain value how many values do you have of that particular domain.

Example from a real life scenario: In a health care company, diagnostic codes were put in the medical claims. The codes had to correspond to different healthcare plans. People were losing so much productivity due to error check messages that they turned off the checks so as to get more claims processed! That was a very bad decision, as they were now putting in junk data. This was quite alright with the operations manager, as he was not measured on the quality of the data, but on how many claims per hour were put in.

Step-2: For each table, column and column value look at how many values are not in my valid domain table. The meta data should specify the data dictionary for every column i.e. the valid values for that column. Any data that is not as per the valid value is a data quality problem either in the meta data or in the data itself.

Step-3: Once the defects are found, go and track them down back to source cause(s).

Point to be noted is that, if at all possible, fix the problem in the source system. People have the tendency of applying fixes in the DWH. This is a wrong i.e. if you are fixing the problems in the DW; you are not fixing the root cause. A crude analogy would clarify the point. If you keep cleaning the lake, and keep on flushing the toilet in the lake, you are not solving the problem. The problem is not being fixed at the source system, therefore, it will persist.

Attribute Domain Validation: What next?

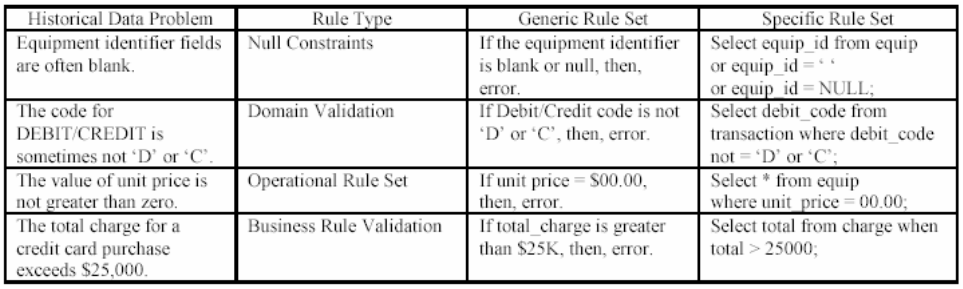
What to do next?

- Trace back to source cause(s).
- Quantify business impact of the defects.
- Assess cost (and time frame) to fix and proceed accordingly.

Easier said than done, this unfortunately is a big problem, as it involves office politics in most organizations. The reason being, operational people do not care about the DWH, they are not measured on data quality, nobody cares. The requirement is to apply pressure on the operational source system personnel to fix the data quality problems, and turn ON the error checks etc.

So what is the solution? The best way of applying pressure is to publish. Publish what is the quality of the data you get from the source system. You will be amazed at the results, how fast people start fixing their data. You can beg all you can behind closed doors, but when it becomes public knowledge activity starts. But have to be careful because of office politics. However, before taking any actions quantify business impact of the defects, and subsequently assess cost (and time frame) for fix and proceed accordingly.

Data Quality Rules

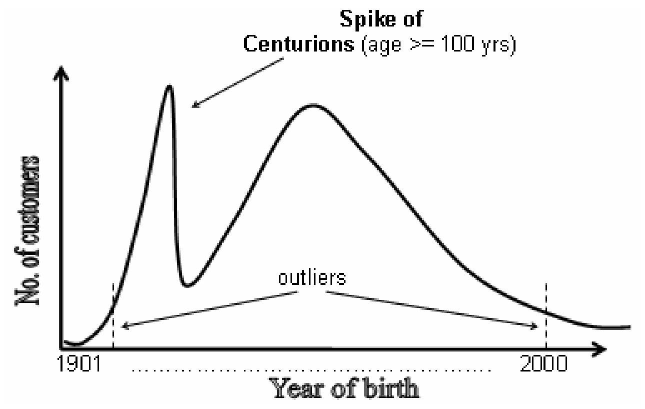


Historical Data Problem	Rule Type	Generic Rule Set	Specific Rule Set
Equipment identifier fields are often blank.	Null Constraints	If the equipment identifier is blank or null, then, error.	Select equip_id from equip or equip_id = ' ' or equip_id = NULL;
The code for DEBIT/CREDIT is sometimes not 'D' or 'C'.	Domain Validation	If Debit/Credit code is not 'D' or 'C', then, error.	Select debit_code from transaction where debit_code not = 'D' or 'C';
The value of unit price is not greater than zero.	Operational Rule Set	If unit price = $00.00, then, error.	Select * from equip where unit_price = 00.00;
The total charge for a credit card purchase exceeds $25,000.	Business Rule Validation	If total_charge is greater than $25K, then, error.	Select total from charge when total > 25000;

Table-22.1: Data Quality Rules

Specific data problems are linked to business rules, and then generic and specific rule sets are established to measure how good the data is within an information system. Table 22.1 illustrates several rule sets and an acceptable method of documenting known data quality problems. Establish a set of rule sets and measurements to execute as SQL statements or as data quality filters in an automated data quality assessment tool. The rule sets represent the data quality metrics used to judge conformance of data to business rules. Data quality project managers use established and relevant data standards as the basis for establishing rule sets. These data standards provide valid values for many common data elements such as Country Code, Country Name, and City Abbreviation.

Statistical Validation using Histogram



NOTE: For a certain environment, the above distribution may be perfectly normal.

Figure-22.1: Statistical Validation using Histogram

To check the accuracy of the date of birth, construct a histogram of date of births. This histogram could be of year of birth or date of birth excluding the year. If the year of birth is missing in a claim, or someone would not like to disclose it while registering online, usually the year of choice is 1901 ("magic" value) or date of birth as "1st Jan". This will result in a huge spike for centurions or New Year birthdays. While you expected an even distribution of birthdays across the year, you will get a spike as shown in Figure22.1.

The approach should be to try all the different ways of constructing histograms and look for large spikes. If there is something wrong, then based on reasonable business knowledge, you are likely to detect it. Note that this is NOT data mining. In data mining you are looking for patterns that you don't expect to exist or checking a hypothesis. Over here you know what you are looking for.

Lecture-23
Total DQM

TDQM in a DWH

Many DW projects do not deliver to full potential because they treat data quality as a one-time undertaking as part of user acceptance testing (UAT) and then forgetting about it. It is very important that data quality management is undertaken as a continuous improvement process...not a one-shot deal!

Use an iterative approach to achieve data quality. Start by defining the quality measurements, and then take the measurements. Go through the cycle as shown in Figure 23.1, this will drive you to the stable stage i.e. CMM-5, where you have a small amount of acceptable data quality defects. Here acceptable is based on cost vs. the value.

Data Quality Management Process

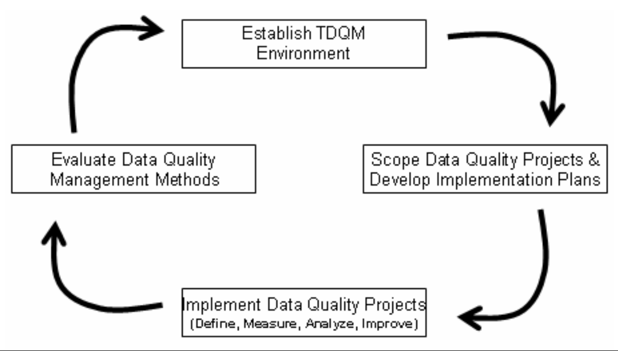

Establish TDQM Environment
Scope Data Quality Projects & Develop Implementation Plans
Implement Data Quality Projects (Define, Measure, Analyze, Improve)
Evaluate Data Quality Management Methods

Figure-23.1: Data Quality Management Process

1. Establish Data Quality Management Environment

Securing a commitment to the Data Quality Management process is accomplished by establishing the data quality management environment between information system project managers and establishing conditions to encourage team work between functional and information system development professionals. Functional users of legacy information systems know data quality problems of the current systems but do not know how to systematically improve existing data. Information system developers know how to identify data quality problems but do not know how to change the functional requirements that drive the systemic improvement of data. Given the existing barriers to communication, establishing the data quality environment involves participation of both functional users and information system administrators.

2. Scope Data Quality Projects & Develop Implementation Plan

For each data quality analysis project selected, the data quality project manager defines the scope of the project and defines the level of analysis that will be the most beneficial for the project under question. Draft an initial plan that addresses the following elements.

- *Task Summary*: Project goals, scope, and potential benefits
- *Task Description:* Describe data quality analysis tasks
- *Project Approach:* Summarize tasks and tools used to provide a baseline of existing data quality
- *Schedule:* Identify task start, completion dates, and project milestones
- *Resources:* Identify resources required to complete the data quality assessment. Include costs connected with tools acquisition, labor hours (by labor category), training, travel, and other direct and indirect costs

3. Implement Data Quality Projects (Define, Measure, Analyze, Improve)

A data quality analysis project consists of four activities. The data quality project manager performs these activities with input from the functional users of the data, system developers, and database administrators of the legacy and target database systems.

- *Define:* Identify functional user data quality requirements and establish data quality metrics.
- *Measure:* Measure conformance to current business rules and develop exception reports.
- *Analyze:* Verify, validate, and assess poor data quality causes. Define improvement opportunities.
- Im*prove:* Select/prioritize data quality improvement opportunities.

Improving data quality may lead to changing data entry procedures, updating data validation rules, and/or use of company data standards to prescribe a uniform representation of data used throughout the company.

4. Evaluate Data Quality Management Methods

The last step in the Data Quality Management process is to evaluate and assess progress made in implementing data quality initiatives and/or projects. All participants in the Data Quality Management process (functional users, program managers, developers, and the Office of Data Management) should review progress with respect to: (1) modifying or rejuvenating existing methods to data quality management and/or (2) determining whether data quality projects have helped to achieve demonstrable goals and benefits. Evaluating and assessing data quality work reinforces the idea that Data Quality Management is not a program, but a new way of doing business.

The House of Quality

In 1972 the Mitsubishi Shipyards in Kobe developed a technique in which customer wants were linked to product specifications via a matrix format as shown in Figure 23.2. This technique is known today as The House of Quality and is one of many techniques of Quality Function Deployment, which can briefly be defined as “a system for translating customer requirements into appropriate company requirements”.

The purpose of the technique is to reduce two types of risk. First, the risk that the product specification does not comply with the wants of the predetermined target group of customers. Secondly, the risk that the final product does not comply with the product specification.

The House of Quality Matrix

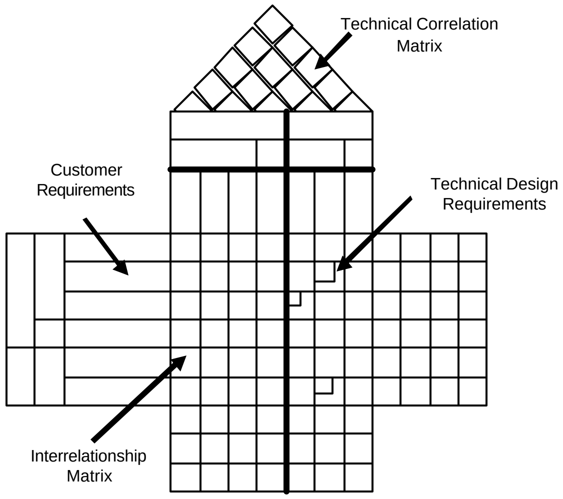


Figure-23.2: The House of Quality Matrix

The House of Quality Data Model

The House of Quality concept was used and introduced with reference to data quality in 1988. The idea is that the end user applications and the data characteristics for those applications such as date of birth etc are tied together, such that for each application tolerances on data quality are placed, including no needs. If the tolerances are not within (say) x%, then the corresponding data can not be used. So we are looking at:

- Establishing specifications for data in the data warehouse.
- Establishing applications for data in the data warehouse.
- Establishing tolerances for conformity to specifications for each combination of applications and data specifications.

If it costs more to deliver the tolerance than the value of the application then the company is not ready to do the application yet. It means the company has to make a business decision not to do the application with dirty data, or to do the application and pretend that the data is of high quality.

The House of Data Quality: Next Step

Getting more sophisticated, we can say that if you give me data with *x%* data quality, the value of the application will be *y* millions of dollars. If you give me *2x%* data quality, then the value of the application is *3y* millions of dollars. Why? Because now the predictive model is more accurate.

If you give me date of birth 90% of the time, then I can make you a million dollars on better customer retention. But if you give me date of birth 95% of the time, then the accuracy goes much higher, and how much is that worth to you? The question is. What is the value of the application, relative to the specific data quality specifications?

How to improve Data Quality?

The four categories of Data Quality Improvement

- Process
- System
- Policy & Procedure
- Data Design

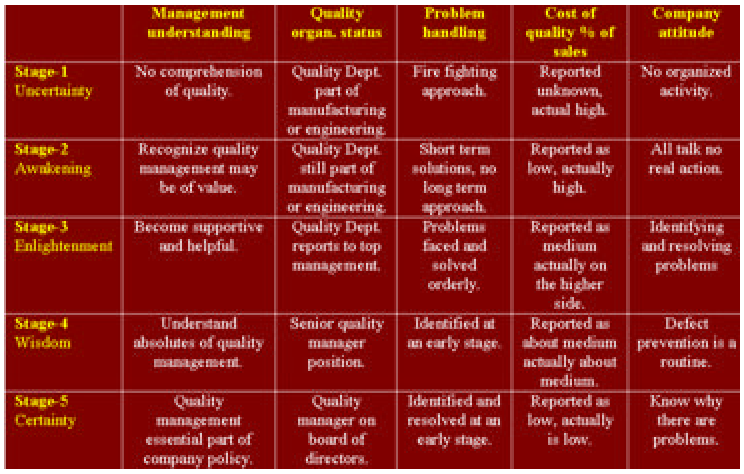
Process Improvement: Improve the functional processes used to create, manage, access, and use data. Functional process changes may encourage centralized data entry, eliminate non-value added activities, and place data quality responsibilities where data is entered into the data set (e.g., certification of data)

System Improvement: Software, hardware, and telecommunication changes can improve data quality. For example, security software can minimize damage done by malicious updates to databases by unauthorized users. Hardware improvements may make batch loads faster and thereby make it unnecessary to turn off edit and validation constraints when loading data to a database. Telecommunications improvements (e.g. increasing bandwidth) may provide easier access to data and improve both the accuracy and timeliness of data. Other system improvements may include updating end user, operation, and maintenance manuals, and providing additional user training.

Policy and Procedure Improvement: Resolve conflicts in existing policies and procedures and institutionalize behaviors that promote good data quality. Develop Standard Operating Procedures for the information system to document the data quality rule sets/filters used to measure data quality. Perform periodic data quality checks as part of the Standard Operating Procedures to increase data quality.

Data Design Improvement: Improve the overall data design and use data standards. Adding primary key constraints, indexes, unique key constraints, triggers, stored functions and procedures, controlling administration of user privileges, enforcing security features, and referential integrity constraints can improve database design.

Quality Management Maturity Grid



	Management understanding	Quality organ. status	Problem handling	Cost of quality % of sales	Company attitude
Stage-1 Uncertainty	No comprehension of quality.	Quality Dept. part of manufacturing or engineering.	Fire fighting approach.	Reported unknown, actual high.	No organized activity.
Stage-2 Awakening	Recognize quality management may be of value.	Quality Dept. still part of manufacturing or engineering.	Short term solutions, no long term approach.	Reported as low, actually high.	All talk no real action.
Stage-3 Enlightenment	Become supportive and helpful.	Quality Dept. reports to top management.	Problems faced and solved orderly.	Reported as medium actually on the higher side.	Identifying and resolving problems
Stage-4 Wisdom	Understand absolutes of quality management.	Senior quality manager position.	Identified at an early stage.	Reported as about medium actually about medium.	Defect prevention is a routine.
Stage-5 Certainty	Quality management essential part of company policy.	Quality manager on board of directors.	Identified and resolved at an early stage.	Reported as low, actually is low.	Know why there are problems.

Table-23.1: Quality Management Maturity Grid

Table 23.1 shows the quality management maturity grid. Each of the five stages will be discussed in detail.

Stage 1: *Uncertainty*

Stage-1 is denial. People don't think data quality is an issue. It is the responsibility of the people collecting the data. It is their problem. There is no data quality inspection; nobody actually cares about the data quality in the processes. The problems are handled on a fire fighting basis, you discover problems (or they pop up) and you fix them using a band aid approach and the problems are rarely resolved. There are a large number of companies in this stage. Some may not even have a quality department, or even when they have one, not paying enough attention and committing resources.

Stage 2: *Awakening*

People think that data quality management does have a value. Management talks about it, but does nothing. This is like all talk and no action stage. People go around giving presentations, but really there is no action, just motivational stuff and not doing any thing. If there is a major problem, the approach is to put together an ad hoc team, that team will attack and resolve the problem. In the end they say that somebody messed up.

Stage 3: *Enlightenment*
A light goes on, may be there is a problem. The quality department starts reporting instead of hiding the problems in the closet. You actually start to measure data quality and are reasonably close. Actually the management is now committed, and have invested resources.

Stage 4: *Wisdom*

Value of quality is seen as something that is needful and meaningful. Data quality starts coming on the performance evaluation of employees i.e. what are they doing for data quality. There is a

14 step program for continuous improvement. They actually start going through it, and start taking proactive actions to prevent data quality defects from happening. So rather than fixing problems AFTER they have occurred, you start taking steps BEFORE the problems occur i.e. practicing prevention. Quality comes from prevention. And prevention is a result of training, discipline, example, leadership, and more. So there is a fundamental shift. Now you are looking at organizational processes, and changing them.

How do you know an organization is in stage 4? They should have looked at their data quality processes and found out why data quality problems are reoccurring and fixed them. If they are still discovering data quality problems, then certainly they are not in stage-4.

There are actually a very small number of organizations in stage-4. Usually they have a DW in place and discovered how bad the quality of their data is before they start moving on this maturity grid.

Stage 5: *Certainty*

The approach is "I know about the quality of my data" and/or "I am measuring it for real". Quality improvement is NOT something that is done on ad-hoc basis, but is part of everything. People know why they are doing it, and why they do not have data quality problems.

Misconceptions on Data Quality

1. You Can Fix Data
2. Data Quality is an IT Problem

1. You Can Fix Data

Fixing implies that there was something wrong with the original data, and you can fix it once and be done with it. In reality, the problem may have been not with the data itself, but rather in the way it was used. When you manage data you manage data quality. It's an ongoing process. Data cleansing is not the answer to data quality issues. Yes, data cleansing does address some important data quality problems has and offers a solid business value ROI, but it is only *one* element of the data quality puzzle. Too often the business purchases a data cleansing tool and thinks the problem is solved. In other cases, because the cost of data cleansing tools is high, a business may decide that it is too expensive for them to deal with the problem.

2. Data Quality is an IT Problem

Data quality is a *company* problem that costs a business in many ways. Although IT can help address the problem of data quality, the business has to own the data and the business processes that create or use it. The business has to define the metrics for data quality - its completeness, consistency, relevancy and timeliness. The business has to determine the threshold between data quality and ROI. IT can enable the processes and manage data through technology, but the business has to define it. For an enterprise-wide data quality effort to be initiated and successful on an ongoing basis, it needs to be truly a joint business and IT effort.

Misconceptions on Data Quality

3. All) Problem is in the Data Sources or Data Entry
4. The Data Warehouse will provide a single source of truth
5. Compare with the master copy will fix the problem

3. The Problem is in the Data Sources or Data Entry

Data entry or operational systems are often blamed for data quality problems. Although incorrectly entered or missing data is a problem, it is far from the only data quality problem. Also, everyone blames their data quality problems on the systems that they sourced the data from. Although some of that may be true, a large part of the data quality issue is the consistency, relevancy and timeliness of the data. If two divisions are using different customer identifiers or product numbers, does it mean that one of them has the wrong numbers or is the problem one of consistency between the divisions? If the problem is consistency, then it is an *enterprise* issue, not a *divisional* issue. The long-term solution may be for all divisions to use the same codes, but that has to be an enterprise decision.

4. The Data Warehouse will provide a Single Version of the Truth

In an ideal world, every report or analysis performed by the business exclusively uses data sourced from the data warehouse - data that has gone through data cleansing and quality processes and includes constant interpretations such as profit or sales calculations. If everyone uses the data warehouse's data exclusively and it meets your data quality metrics then it is the single version of the truth.

However, two significant conditions lessen the likelihood that the data warehouse solves your data quality issues by itself. First, people get data for their reports and analysis from a variety of data sources - data warehouse (sometimes there are multiple data warehouses in an enterprise), data marts and cubes (that you hope were sourced from the data warehouse). They also get data from systems such as ERP, CRM, and budgeting and planning systems that may be sourced into the data warehouse themselves. In these cases, ensuring data quality in the data warehouse alone is not enough. Multiple data silos mean multiple versions of the truth and multiple interpretations of the truth. Data quality has to be addressed across these data silos, not just in the data warehouse.

Second, data quality involves the source data and its transformation into information. That means that even if every report and analysis gets data from the same data warehouse, if the business transformations and interpretations in these reports are different then there still are significant data quality issues. Data quality processes need to involve data creation; the staging of data in data warehouses, data marts, cubes and data shadow systems; and information consumption in the form of reports and business analysis. Applying data quality to the dat a itself and not its usage as information is not sufficient.

Normally source system reconciliation is used to check if the ETL is working properly. This technique is useful when you are doing test and validation of the ETL, but is not recommended on a long term basis. This will only be done when major changes are made in the ETL. Since the process is manual, it is human intensive, so it will be very expensive and inefficient.

All the other four techniques discussed in the previous sections can be automated by writing scripts, and running them every month to collect statistics, generating reports, and identifying problems and subsequently noting the progress of quality control. But one has to be very clever to write scripts for source system reconciliation. It can turn out to be a nightmare. The main problems are that you may have garbage in the DWH, garbage in the legacy system, and that does not really tell you about the data quality. Recall Orr's Law #4 - "Data quality problems increase with the age of the system!"

TDQM: Successful Data Sourcing

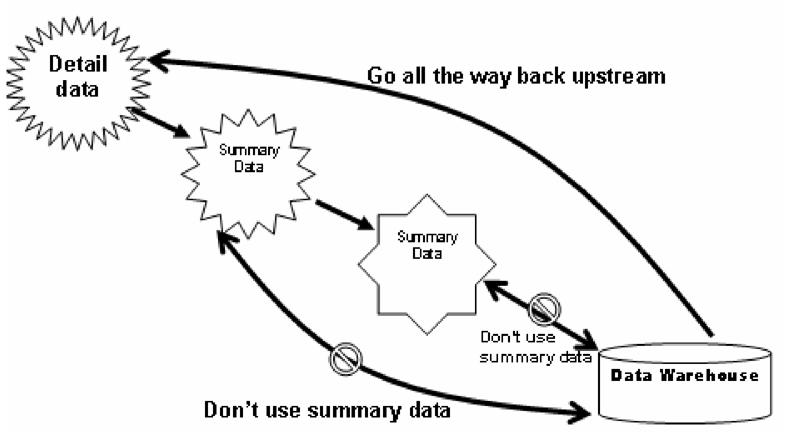


Figure-23.3: TDQM: Successful Data Sourcing

If you take data from downstream data sources it's likely that not only you are going to accumulate the data quality defects of the upstream sources, but also the compromises that the downstream may have added, especially the summarizations. Once you summarize, you can not trace back to detailed data. Go as high upstream for the gold copy of the data as possible as shown in Figure 23.3.

You also have to be aware of the synchronization issues. You take the transaction data and the accounts data and there are some transactions that don't exist in the accounts yet, so you have to be careful of the order in which you take the data for synchronization. Synchronization is probably the biggest reason we have referential integrity problems in a DWH i.e. lack of synchronization between different extracts of the source system.

Misconceptions on Data Quality

It came from the legacy system so must be correct.

Another misconception is, "It came from the production transaction processing environment, so it must be correct." However the reality is, the elements required for decision support are often quite different than those required for transaction processing. The objective of OLTP systems is to ensure that all the sales data is in, the numbers sum up correctly, accounts are balanced etc. If the gender data for some (or even most) sales is not present, it does not change the sales dollars. Why bother? Why not enter some junk data in the gender field? This may be fine from the operational point of view, but critical for decision support, when you are interested in determining your market segmentation.

Redefines and other embedded logic in legacy programs often puts data quality delivered to data warehouse in severe question. Y2K is a good example.

Lecture-24
Need for Speed: Parallelism

Data warehouses often contain large tables and require techniques both for managing these large tables and for providing good query performance across these large tables.

Parallel execution dramatically reduces response time for data-intensive operations on large databases typically associated with Decision Support Systems (DSS) and data warehouses. You can also implement parallel execution on certain types of online transaction processing (OLTP) and hybrid systems.

Parallel execution is sometimes called parallelism. Simply expressed, parallelism is the idea of breaking down a task so that, instead of one process doing all of the work in a query, many processes do part of the work at the same time. An example of this is when four processes handle four different quarters in a year instead of one process handling all four quarters by itself. The improvement in performance can be quite high. In this case, data corresponding to each quarter will be a partition, a smaller and more manageable unit of an index or table.

When to parallelize?

Useful for operations that access significant amounts of data.

Useful for operations that can be implemented independent of each other "Divide-&-Conquer"

Parallel execution improves processing for:

- Size — Large table scans and joins
- Size — Creation of large indexes
- D&C — Partitioned index scans
- Bulk inserts, updates, and deletes
- Size — Aggregations and copying
- D&C —

Every operation can not be parallelized, there are some preconditions and one of them being that the operations to be parallelized can be implemented independent of each other. This means that there will be no interference between the operations while they are being parallelized. So what do we gain out of parallelization; many things which can be divided into two such as with reference to size and with reference to divide and conquer. Note that divide and conquer means that we should be able to divide the problem and then solve it and then compile the results i.e. conquer. For example in case of scanning a large table every row has to be checked, in such a case this can be done in parallel thus reducing the overall time. There can be and are many examples too.

Are you ready to parallelize?

Parallelism can be exploited, if there is…

- Symmetric multi-processors (SMP), clusters, or Massively Parallel (MPP) systems AND
- Sufficient I/O bandwidth AND
- Underutilized or intermittently used CPUs (for example, systems where CPU usage is typically less than 30%) AND
- Sufficient memory to support additional memory-intensive processes such as sorts, hashing, and I/O buffers

Word of caution

Parallelism can reduce system performance on over-utilized systems or systems with small I/O bandwidth.

One can not just get up and parallelize, there are certain hardware and software requirements other than the nature of the problem itself. The first and the foremost being a multiprocessor environment which could consist of a small number of high performance processors to large number of not so fast machines. Then you need bandwidth to port data to those processors and exchange data between them. There are other options too such as a grid of those machines in the computer lab that are not working during the night or are idling running just screen savers and the list goes on.

A word of caution: Parallelism when not observed or practices carefully can actually degrade the performance, in case the system is over utilized and the law of diminishing returns sets in or there is insufficient bandwidth and it actually becomes the bottleneck and chokes the system.

Scalability – Size is NOT everything

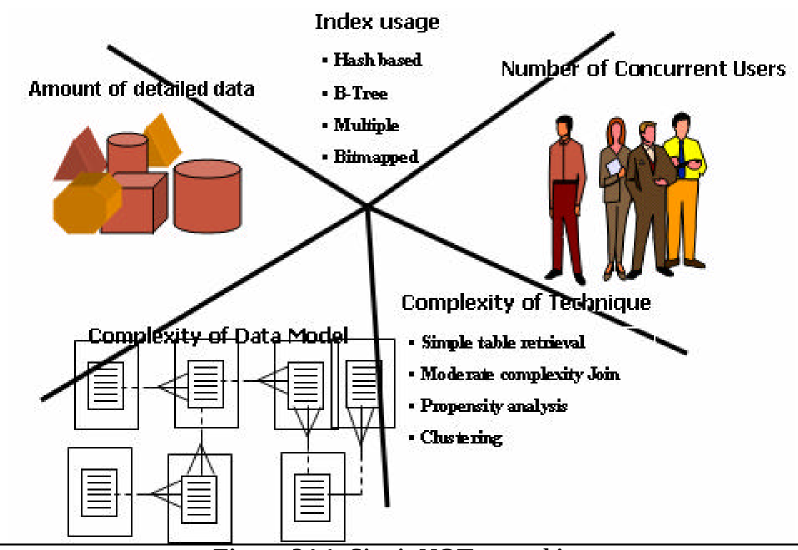
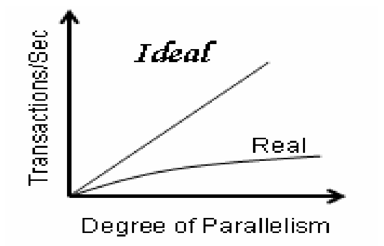


Figure-24.1: Size is NOT everything

It is common today for vendors and customers to boast about the size of their data warehouses. And, in fact, size does matter. But it is not size alone as shown in Figure 24.1. Rather, significant increases in data volume amplifies the issues associated with increasing numbers and varieties of users, greater complexity and richness in the data model, and increasingly sophisticated and complex business questions. Scalability is also about providing great flexibility and analysis potential through richer, albeit more complex, data schemas. Finally, it is just as important to allow for the increasing sophistication of the data warehouse users and evolving business needs. Although the initial use of many data warehouses involves simple, canned, batch reporting many companies are rapidly evolving to far more complex, ad hoc, iterative (not repetitive) business questions – "any query, any time" or "query freedom".

Scalability- Terminology

Speed-Up
More resources means proportionally less time for given amount of data.



Scale-Up
If resources increased in proportion to increase in data size, time is constant

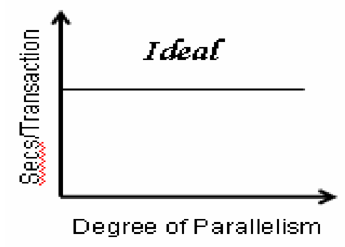


Figure-24.2: Scalability- Terminology

Its time for a reality check. It seems that increasing the processing power in terms of adding more processors or computing resources should give a corresponding speedup are not correct. Ideally this should be true, but in reality the entire problem is hardly parallelizable, hence the speedup is non-linear. Similar behavior is experienced when the resources are increased in proportions to the increase in problem size so that the time for a transaction remains same, ideally this is true, but the reality is different. Why these things happen will become clear when we discuss Amdahl's Law.

Quantifying Speed-up

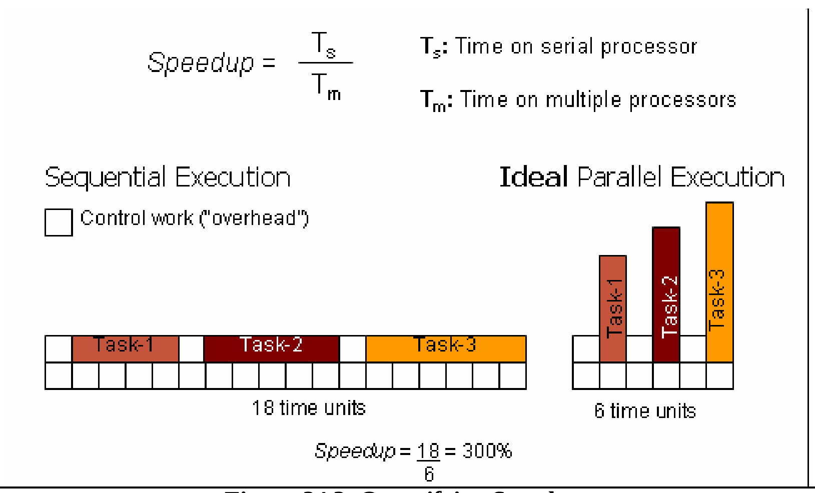

Speedup = $\frac{T_s}{T_m}$
T_s: Time on serial processor
T_m: Time on multiple processors
Sequential Execution
Control work ("overhead")
Task-1
Task-2
Task-3
18 time units
Ideal Parallel Execution
Task-1
Task-2
Task-3
6 time units
Speedup = $\frac{18}{6}$ = 300%

Figure-24.3: Quantifying Speed-up

Data dependencies between different phases of computation introduce synchronization requirements that force sequential execution. Moreover, there is a wide range of capabilities available in commercially implemented software in regard to the level of granularity at which parallelism can be exploited.

As shown in figures 24.2 and 24.3, , the goal of ideal parallel execution is to completely parallelize those parts of a computation that are not constrained by data dependencies. The smaller the portion of the program that must be executed sequentially (s), the greater the scalability of the computation.

Speed-Up & Amdahl's Law

Reveals maximum expected speedup from parallel algorithms given the proportion of task that must be computed sequentially. It gives the speedup S as

$$S \leq \frac{1}{f + (1 - f)/N}$$

f is the fraction of the problem that must be computed sequentially
N is the number of processors

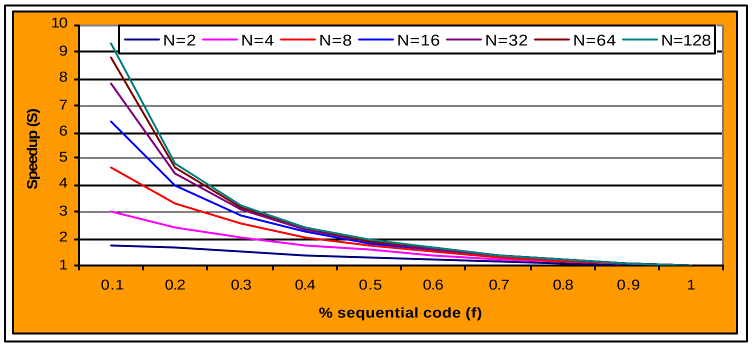
As *f* approaches 0, S approaches N

$$\left.\begin{array}{l}\text{Example-1: } f = 5\% \text{ and } N = 100 \text{ then } S = 16.8\\ \text{Example-2: } f = 10\% \text{ and } N = 200 \text{ then } S = 9.56\end{array}\right\}\ \text{Not 1:1 Ratio}$$

The processing for parallel tasks can be spread across multiple processors. So, if 90% of our processing can be parallelized and 10% must be serial we can speed up the process by a factor of 3.08 when we use four independent processors for the parallel portion. This example also assumes 0 overhead and "perfect" parallelism. Thus, a database query that would run for 10 minutes when processed serially would, in this example, run in 2.63 minutes (10/3.08) when the parallel tasks were executed on four independent processors.

As you can see, if we increase the overhead for parallel processing or decrease the amount of parallelism available to the processors, the time it takes to complete the query will increase according to the formula above.

Amdahl's Law: Limits of parallelization



For less than 80% parallelism, the speedup drastically drops.

At 90% parallelism, 128 processors give performance of less than 10 processors.

Figure-24.4: Amdahl's Law

As we can see in the graphical representation of Amdahl's Law as shown in Figure24.4, the realized benefit of additional processors is significantly diminishes as the amount of sequential processing increases. In this graph, the vertical axis is the system speed-up. As the overall percentage of sequential processing increases (with a corresponding decrease in parallel processing) the relative effectiveness (utilization) of additional processors decreases. At some point, the cost of an additional processor actually exceeds the incremental benefit.

Parallelization OLTP Vs. DSS

There is a big difference.

DSS

Parallelization of a SINGLE query

OLTP

Parallelization of MULTIPLE queries
Or Batch updates in parallel

During business hours, most OLTP systems should probably not use parallel execution. During off-hours, however, parallel execution can effectively process high-volume batch operations. For example, a bank can use parallelized batch programs to perform the millions of updates required to apply interest to accounts.

The most common example of using parallel execution is for DSS. Complex queries, such as those involving joins or searches of very large tables, are often best run in parallel.

Brief Intro to Parallel Processing

- Parallel Hardware Architectures
  - Symmetric Multi Processing (SMP)
  - Distributed Memory or Massively Parallel Processing (MPP)
  - Non-uniform Memo ry Access (NUMA)

- Parallel Software Architectures
  - Shared Memory
  - Shard Disk
  - Shared Nothing

  $\left.\begin{array}{l}\text{Shared Memory}\\\text{Shard Disk}\end{array}\right\}$ Shared everything

- Types of parallelism
  - Data Parallelism
  - Spatial Parallelism

NUMA

Usually on an SMP system, all memory beyond the caches costs an equal amount to reach for each CPU. In NUMA systems, some memory can be accessed more quickly than other parts, and thus called as Non-Uniform Memory Access. This term is generally used to describe a shared-memory computer containing a hierarchy of memories, with different access times for each level in the hierarchy. The distinguishing feature is that the time required to access memory locations is not uniform i.e. access times to different locations can be different.

Symmetrical Multi Processing (SMP)

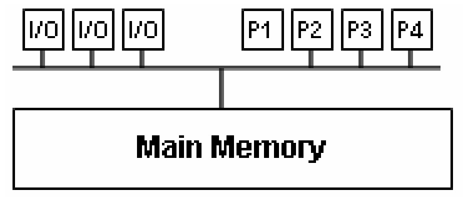


Figure-24.5: Symmetrical Multi Processing

SMP (Symmetric Multiprocessing) is a computer architecture that provides fast performance by making multiple CPUs available to complete individual processes simultaneously (multiprocessing). Unlike asymmetrical processing, any idle processor can be assigned any task, and additional CPUs can be added to improve performance and handle increased work load. A variety of specialized operating systems and hardware arrangements are available to support SMP. Specific applications can benefit from SMP if the code allows multithreading.

SMP uses a single operating system and shares common memory and disk input/output resources. Both UNIX and Windows NT support SMP.

Distributed Memory Machines

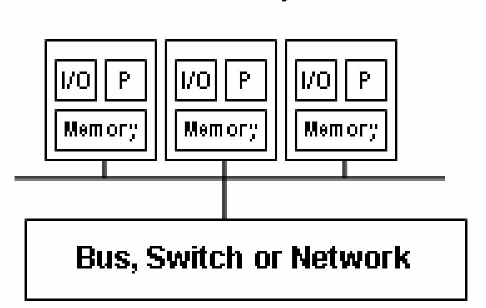


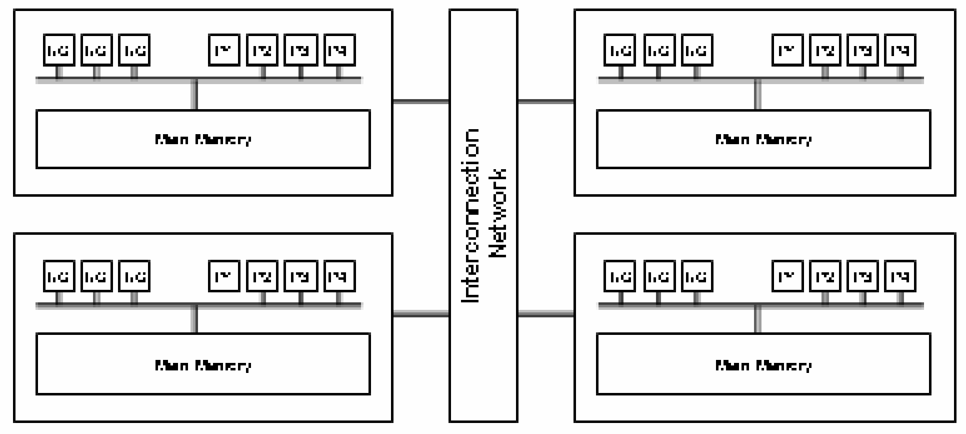
Figure-24.6: Distributed Memory Machines

Special-purpose multiprocessing hardware comes in two flavors i.e. shared memory and distributed memory machines. In a shared-memory machine, all processors have access to a common main memory. In a distributed-memory machine, each processor has its own main memory, and the processors are connected through a sophisticated interconnection network. A collection of networked PCs is also a kind of distributed-memory parallel machine.

Communication between processors is an important prerequisite for all but the most trivial parallel processing tasks (thus bandwidth can become a bottleneck). In a shared-memory machine, a processor can simply write a value into a particular memory location, and all other processors can read this value. In a distributed-memory machine, exchanging values of variables involves explicit communication over the network, thus need for a high speed interconnection network.

Distributed Shared Memory Machines

A little bit of both worlds!



It is also known as *Virtual Shared Memory.* This memory model is the attempt of a compromise between shared und distributed memory. The distributed memory has been combined with an OS-based message passing system which simulates the presence of a global shared memory, e.g., KSR: "Sea of addresses" and SGI: "Interconnection fabric". The plus side is that a sequential code will run immediately on that memory model. If the algorithms take advantage of the local properties of data (i.e., most data accesses of a process can be served from its own local memory) then a good scalability will be achieved.

Figure-24.7: Distributed Shared Memory Machines

Shared disk RDBMS Architecture

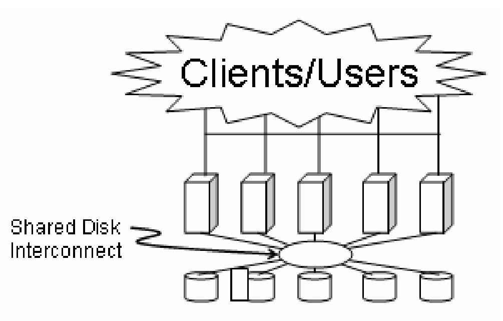


Figure-24.7: Shared disk RDBMS Architecture

Shared disk database architecture as shown in Figure 24.7 is used by major database vendors. The idea here is that all processors have equal access to data stored on disk. It does not mater if it is a local disk or a remote disk, it is treated a single logical disk all the time. So we rely on high bandwidth inter-processing communication to ship disk blocks to the requesting processor. This approach allows multiple instances to see the same data. Every database instance sees everything. Note that database instances mean different things for different databases. When I say database instances in this context, it means collection of processes and threads that all share the same database buffer cash. In the shared disk approach, transactions running on any instance can directly read or modify any part of the database. Such systems require the use of inter-node communication to synchronize update activities performed from multiple nodes. When two or more nodes contend for the same data block, the node that has a lock on the data has to act on the data and release the lock, before the other nodes can access the same data block.

Advantages:

A benefit of the shared disk approach is it provides a high level of fault tolerance with all data remaining accessible even if there is only one surviving node.

Disadvantages:

Maintaining locking consistency over all nodes can become a problem in large clusters. So I can have multiple database instances each with it's own database buffer cache all accessing the same set of disk blocks. This is a shared everything disk architecture. Now if multiple database instances are accessing the same tables and same blocks, then some locking mechanism will be required to maintain database buffer cash coherency. Because if a data block is in the buffer cache of P1 and the same data block is in the buffer cash of P2 then there is a problem. So there is something called distributed lock management that has to be implemented to maintain coherency between the databases buffer cashes across these different database instances.

And that leads to a lot of performance issues in shared everything databases because every time when lock management is performed, it becomes serial processing. There are two approaches to solving this problem i.e. hardware mechanism and a software mechanism. In the hardware mechanism, a coupling facility is used. The coupling facility manages all the locks to control coherency in the database buffer cash. Another vendor took a different approach; because it sells a more portable database that runs on any platform, therefore, it couldn't rely on special hardware. Therefore, there is a software lock management system called the distributed lock manager, which is used to mange across different database instances. In most cases both techniques must guarantee that there is never incoherency of data blocks across database instances.

Shared Nothing RDBMS Architecture

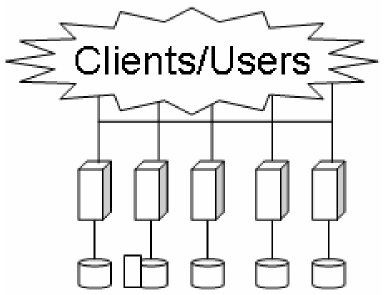


Figure-24.8: Shared Nothing RDBMS Architecture

In case of shared nothing architecture as shown in Figure 24.8, there is no lock contention and therefore any time you have locking problem then you also have serialization issue. The idea is that each database table partition in the database instances e.g. the customer table and Order table exist on all the database instances. So the parallelism is really already built in. There is never any confusion and there is never any locking problem. If we join two tables with the same partitioning column, and the partitioning was performed using hash partitioning, then that is a local join and is very efficient.

A request will be made to the "owning" database instance to send the desired columns (projection) from qualifying rows of the source table when data is required by one database instance that is partitioned to a different database instance. In the function shipping approach, the column and row filtering is performed locally by the "owning" database instance so that the amount of information communicated to requesting database instance is only what is required. This is different than in shared disk database architectures where full data blocks (no filtering) are shipped to the requesting database instance.

Advantages:

This works fine in environments where the data ownership by nodes changes relatively infrequently. The typical reasons for changes in ownership are either database reorganizations or node failures.

There is no overhead of maintaining data locking across the cluster

Disadvantages

The data availability depends on the status of the nodes. Should all but one system fail, then only a small subset of the data is available.

Data partitioning is a way of dividing your tables etc. across multiple servers according to some set of rules. However, this requires a good understanding of the application and its data access patterns (which may change).

Shared disk Vs. Shared Nothing RDBMS

- Important note: Do not confuse RDBMS architecture with hardware architecture.
- Shared nothing databases can run on shared everything (SMP or NUMA) hardware.
- Shared disk databases can run on shared nothing (MPP) hardware.

Now a very important point here is not to confuse the software architecture with the hardware architecture. And there is lots of confusion on that point. People think that shared nothing database architectures can only be implemented on shared nothing hardware architectures, that's not true. People think that shared everything database architectures can only be implemented on shared everything hardware architecture, which is not true either. So for example shared nothing database like Teradata can work on an SMP machine, that's not a problem. Because the software is shared nothing that does not mean that the hardware has to be shared nothing. SMP is symmetric multi processing, shared memory, shared bus structure, shared I/O system and so on, it is not a problem. In fact Informix is a shared nothing database architecture which was originally implemented on a shared everything hardware architecture which is an SMP machine.

So shared disk databases some times called shared everything databases are also run on shared nothing hardware. Oracle is a shared everything database architecture and the original implementation of the parallel query feature was written on machine called the N-Cube machine. N-Cube machine is an MPP machine that is a shared nothing hardware architecture but that has a shared everything database. In order to do that, a special layer of software called the VSD (Virtual shared disk) is used. So when an I/O request is made, in a shared everything database environment like ORACLE, every instance of the database can see every data block. If it is a shared nothing environment how do I see other data blocks? With a basically an I/O device driver which looks at the I/O request and if it is local, it says ok access it locally, if it is remote, it ships the I/O request to another Oracle instance it does the I/O for me and then it ships the data back.

Shared Nothing RDBMS & Partitioning

Shared nothing RDBMS architecture requires a static partitioning of each table in the database.

How do you perform the partitioning?

- Hash partitioning
- Key range partitioning.
- List partitioning.
- Round-Robin
- Combinations (Range-Hash & Range-List)

Range partitioning maps data to partitions based on ranges of partition key values that you establish for each partition. It is the most common type of partitioning and is often used with dates. For example, you might want to partition sales data into monthly partitions.

Most shared nothing RDBMS products use a hashing function to define the static partitions because this technique will yield an even distribution of data as long as the hashing key is relatively well distributed for the table to be partitioned. Hash partitioning maps data to partitions based on a hashing algorithm that database product applies to a partitioning key identified by the DBA. The hashing algorithm evenly distributes rows among partitions, giving partitions approximately the same size. Hash partitioning is the ideal method for distributing data evenly across devices. Hash partitioning is a good and easy-to-use alternative to range partitioning when data is not historical and there is no obvious column or column list where logical range partition pruning can be advantageous.

List partitioning enables you to explicitly control how rows map to partitions. You do this by specifying a list of discrete values for the partitioning column in the description for each partition. This is different from range partitioning, where a range of values is associated with a partition and with hash partitioning, where you have no control of the row-to-partition mapping. The advantage of list partitioning is that you can group and organize unordered and unrelated sets of data in a natural way.

Round robin is just like distributing a deck of cards, such that each player gets almost the same number of cards. Hence it is "fair".

Lecture-25
Need for Speed: Hardware Techniques

Data Parallelism: Concept

- Parallel execution of a single data manipulation task across multiple partitions of data.
- Partitions static or dynamic
- Tasks executed almost-independently across partitions.
- "Query coordinator" must coordinate between the independently executing processes.

So data parallelism is I think the simplest form of parallelization. The idea is that we have parallel execution of single data operation across multiple partitions of data. So the idea here is that these partitions of data may be defined statically or dynamically fine, but we are requiring the same operator across these multiple partitions concurrently. And this idea actually of data parallelism has existed for a very long time. So the idea is that you are getting parallelization because we are getting semi-independent execution, data manipulation across the partitions. And as long as we keep the coordination required, we can get very good speedups. Well again this query coordinator, the thing that keeps the query distributed but still working and then collects its results.. Now that query coordinator can potentially be a bottleneck, because if it does too much work, that is serial execution. So the query coordination has to be very small amount of work. Otherwise the overhead gets higher and the serialization of the workload gets higher.

Data Parallelism: Example

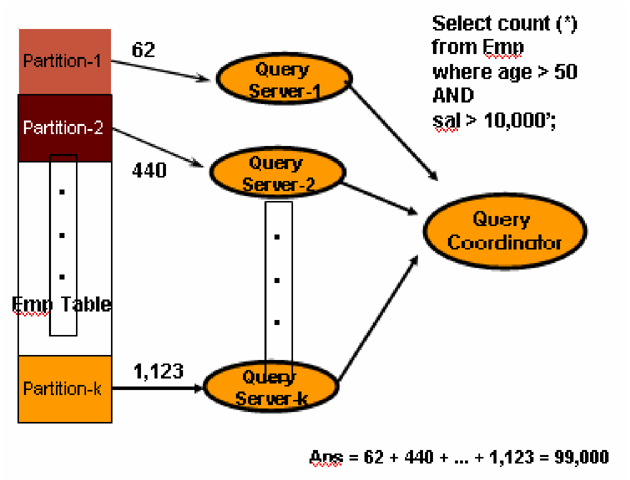


Figure-25.1: Example of Data Parallelism

Suppose that we have a question that we want to select the number of accounts where balance is greater that 5,000$ and the open data is after first of June 2000. This account table and what we end up doing is say ok send the query to each query server and each query server then runs the query against a particular partition as shown in Figure 25.1. And then how many query servers we need and each gets a partial result. So in the data blocks that are processed 1,235 that meet up the criteria. And here we get 536 and here this one found 2,016. The query coordinator just takes these numbers and lets just say there is a 100-degree of parallelism, so that I take 100 numbers and I add them and I get the result. Remember that a query coordinator is software.

Data Parallelism: Ensuring Speed-UP

To get a speed-up of N with N partitions, it must be ensured that:

- There are enough computing resources.
- Query-coordinator is very fast as compared to query servers.
- Work done in each partition almost same to avoid performance bottlenecks.
- Same number of records in each partition would not suffice.
- Need to have uniform distribution of records w.r.t. filter criterion across partitions.

Linear speed up with data parallelism should be achieved as long as the coordinator is not burdened by too much work and the workload is distributed relatively evenly to the participating query servers. If the coordinator does too much work, then the serial processing involved in final results construction will significantly impact overall performance (remember Amdahl's Law!).

Depending on CPU speed and RDBMS efficiency, the number of partitions may be either greater or less than (or equal to) the number of data partitions in the parallel execution of a query. Faster CPUs and more efficient (light weight) query server implementations will generally have more partitions than CPUs (so that more overlap can take place with computational and I/O processing). Less efficient (heavy weight) query server implementations or slow CPUs will tend toward fewer data partitions than CPUs. There is usually a one-to-one correspondence of query servers to data partitions.

Note: You should not confuse data partitions used for the purposes of data parallelism with range partitions used to divide up large tables for ease of management - they are two different things!

The first thing is that the amount of work done by the query coordinator should be very small. It should be small because that work is serial. And if it is serial it will kill the performance. So it should be almost zero.

The second thing is that the workload performed in each partition should be relatively equal. If one query server has much more work to do then any other query server then every body waits for the smothered storm. So in parallel execution environment distribution of work is very important. Every database that is serious about parallel query capability should have data parallelism in the market place today. This is sort of cost of playing the game. You have to at least be able to do this, to be considered as a serious competitor.

Spatial Parallelism (pipelining)

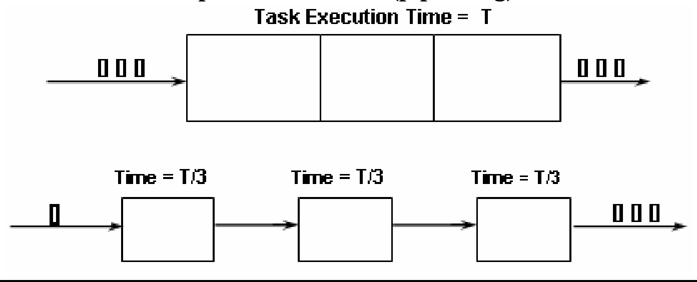


Figure-25.2: Spatial Parallelism (pipelining)

The key idea behind pipeline parallelism is to take a "big" task and break it into subtasks that can be processed concurrently on a stream of data inputs in multiple, overlapping stages of execution.

Pipeline parallelism is the third form of parallel execution. This involves taking a complex task and breaking it down in to sub-tasks. And I would like to use an analogy to describe this because people some times get afraid about pipeline parallelism What I argue is that everybody inherently known what pipeline parallelism is, they just don't know what by that mean.

Pipelining: Time Chart

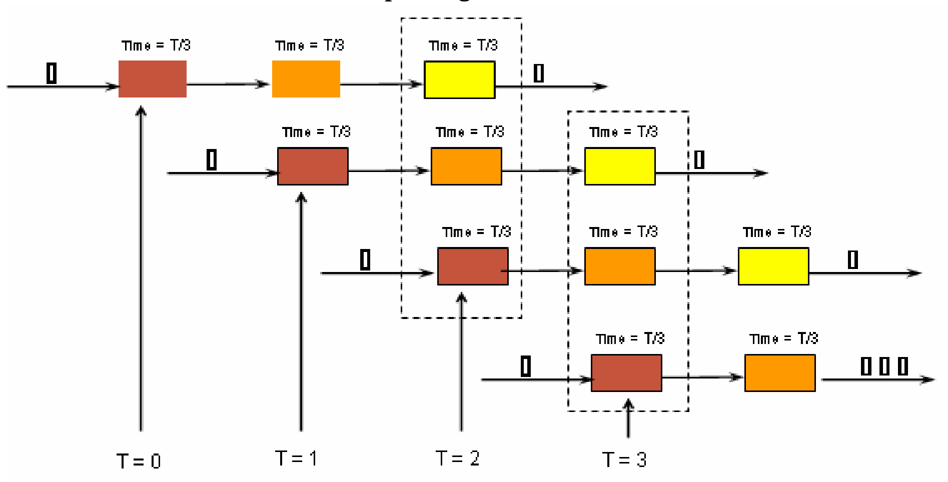


Figure-25.3: Pipelining: Time Chart

Pipelining: Speed-Up Calculation

Time for sequential execution of 1 task = T
Time for sequential execution of N tasks = N * T
(Ideal) time for pipelined execution of one task using an M stage pipeline = T
(Ideal) time for pipelined execution of N tasks using an M stage pipeline = T + ((N-1) × (T/M))

Speed-up (S) = $$S = \frac{NT}{T + (N-1) \times \frac{T}{M}}$$

Pipeline parallelism focuses on increasing throughput of task execution, NOT on decreasing sub-task execution time.

If we take a "big" task that takes T time units to execute on a single data item, then execution of this task on N data items will take N*T time units.

If we break the big task into M equal sized subtasks then we can overlap execution using the pipeline technique. It will take T time units for the first data item to be processed through the pipeline, but after that we will get one data item every T/M time units (assuming no pipeline overhead).

Notice that the time for processing a single data item (latency) does not change with pipelining - only the rate at which we process a large number of data items (throughput) is enhanced with pipelined execution.

What happens is that as the number of laundry loads increases the speedup factor approaches the number of stages in the pipeline but never quite gets there. Because the first load of laundry has to get through costing me full T. and then every body after that is 3/T. So the more loads of laundry the more amortize the cost of filling the pipeline. So this is my excuse for saving laundry to the end of the month because it is much more efficient all right putting a 100 loads of laundry. Pipeline has a reasonable good speedup factor if you have many stage pipeline and you have a good distribution of work. Now, normally when we think of pipeline as hardware pipeline Especially people who come from architecture background. But this can also be a software pipeline. In databases it is really a software pipeline. So if we go back to the earlier slide and we ask a question about how can I have pipeline in this example? Is that when I join these two together in a merge join. I can take result of that and put in to a hash table at the same time as I am joining rows here. So I am simultaneously doing merges and hashes and producing results then. And it turns out that again all the pipelines have to be relatively balanced to get that work well and it may or may not, I don't know about this particular example, I have not worked that. But you can see theoretically how you can have software pipeline in addition to data and spatial parallelism.

Pipelining: Speed-Up Example

Example: Bottling soft drinks in a factory

10 CRATES LOADS OF BOTTLES
Sequential execution = 10 × T
Fill bottle, Seal bottle, Label Bottle pipeline = T + T × (9-1)/3 = 4 × T
Speed-up = 2.50

20 CRATES LOADS OF BOTTLES
Sequential execution = $20 \times T$
Fill bottle, Seal bottle, Label Bottle pipeline = $T + T \times (20-1)/3 = 7.3 \times T$
Speed-up = 2.72

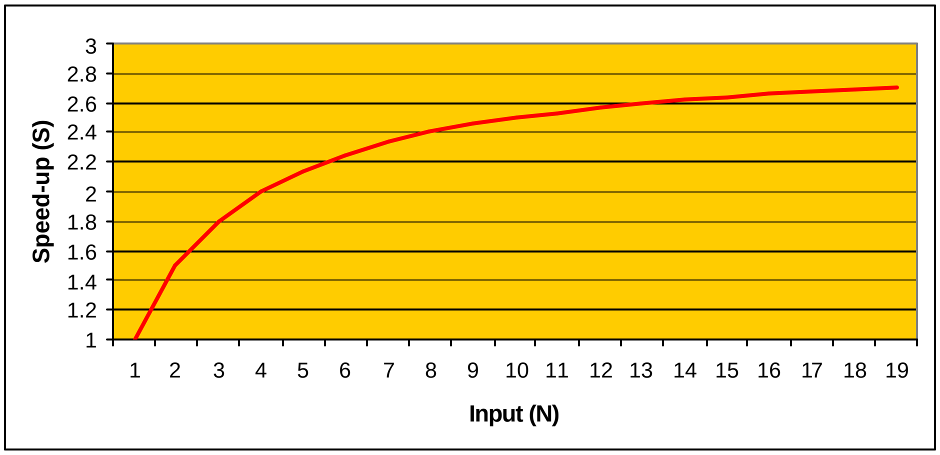
40 CRATES LOADS OF BOTTLES
Sequential execution = $40 \times T$
Fill bottle, Seal bottle, Label Bottle pipeline = $T + T \times (40-1)/3 = 14.0 \times T$
Speed-up = 2.85

Since we have taken the bottling task and divided it into three subtasks (filling, sealing and labeling), we have a three stage (M=3) pipeline.

The speed-up factor is the time it takes to execute in a sequential (non-pipelined) fashion divided by the time it takes to execute with pipelining.

Notice that the speed-up factor gets larger as the number of crates of bottles increases. This is because the start-up cost for the pipeline (getting the first load of bottles through) is amortized over a larger number of laundry "executions" which thereafter are delivered every T/M time units. The maximum speed-up factor achievable is equal to the number of pipeline stages (M=3 in our example).

Pipelining: Input vs. Speed-Up



Asymptotic limit on speed-up for M stage pipeline is M.

The speed-up will NEVER be M, as initially filling the pipeline took T time units.

Figure-25.4: Pipelining: Input vs. Speed-Up

Pipelining: Limitations

Pipeline parallelism is a good fit for data warehousing (where we are working with lots of data), but it makes no sense for OLTP because OLTP tasks are not big enough to justify breaking them down into subtasks.

In order to eliminate overhead in breaking a big task into an M-stage pipeline, it is important that there is efficient synchronization between producer/consumer tasks in the pipeline execution. Also, it is important to avoid any tasks for which all data must be processed before moving on to the next step because this results in pipeline stalls (where we cannot feed the next execution step until all data has been processed through the current execution step). Sort-merge joins "kill" pipeline execution because the sort will stall the pipeline. Hash joins do very well because there is no pipeline stall – provided that you have large amounts of memory available to accommodate all the parallel (in-memory) hashing activity.

Performance is dictated by the slowest stage in the pipeline. Efficient RDBMS implementations will break up the work into pipeline stages that are relatively balanced. Informix XPS is the only RDBMS that has truly mastered pipelined execution in a data warehouse environment - albeit with very high communication costs for synchronization between the producer/consumer tasks.

Partitioning & Queries

- Full Table Scan
- Point Queries:
- Range Queries
- Round Robin
- Hash Partitioning
- Range Partitioning

Parallel Sorting

- Scan in parallel, and range partition on the go.
- As partitioned data becomes available, perform "local" sorting.
- Resulting data is sorted and again range partitioned.
- Problem: skew or "hot spot".
- Solution: Sample the data at start to determine partition points.

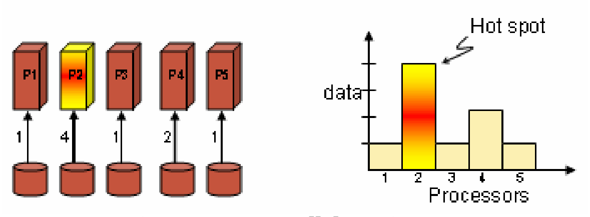


Figure-25.5: Parallel Sorting

Consider figure-25.5 that shows a non-uniform distribution of data that defeats the purpose of a parallel sort as processor-2 develops a hot spot. The solution is to sample the data to determine the data demographics and then partitioning the data so as to come up with near uniform distribution to exploit parallelism.

Skew in Partitioning

- Attribute-value skew.
- Partition skew.

There can be two types of skews i.e. non uniform distribution when the data is distributed across the processors. One type of skew is dependent in the properties of the data, consider the example of data about cancellation of reservations. It is obvious that most cancellations in the history of airline travel occurred during the last quarter of 2001. Therefore, whenever the data is distributed based on date for year 2001 it will be always skewed. This can also be looked at from the perspective of partition skew, as date is typically seen to result in non-uniform distribution of data.

Handling Skew in Range-Partitioning

- Sort
- Construct the partition vector
- Duplicate entries or imbalances

There are number of ways to handle the skew in the data when it is partitioned based on the range, here date is a good example with data distributed based in quarters across four processors. One solution is to sort the data this would identify the “clusters” within the data, then bases on them more or less equal partitions could be created resulted in elimination or reduction of sekw.

Barriers to Linear Speedup & Scale-up

- Amdahl’ Law
- Startup
- Interference
- Skew

As I said in the first lecture on parallelism, intuitively we feel that as the number of processors increase, the speedup should also increase. Thus theoretically there should be a linear speedup. However, in reality this is not the case. The biggest hurled is the Amdahl's law which we have discussed in detail. The other problem is the startup cost, it takes a while for the system to get started and that time when amortized over the entire processing time results in a less than a linear speedup. Then is the interference among different processes or the dependencies among the processes or some operations within the problem itself such as empting of the pipeline, this results in degradation of performance.

Finally the skew in the data. Parallelization is based on the premise that there is a full utilization of the processors and all of them are bust most or all of the time. However, if there is a skew in the partitioning of data i.e. a non-uniform distribution, then some of the processors will be working while other will be idle. And the processor that takes the mosst time (which has the most data too) will become the bottleneck.

Lecture-26
Conventional Indexing Techniques

We start out with what we call the conventional or standard indexing techniques and then will get in to more advanced techniques, typical of a DSS environment. So we will look at conventional indexing and one of the things that I want to do is challenge your traditional way of thinking about indices. Again it goes back to this difference between transaction processing workloads and decision support workloads or traditional way of thinking is based on transaction processing and it dose not work in a decision support environment. So we need to have a different way of thinking about how to apply indexing in this type of model. So we will talk about B-Tree indexing and hash indexing and move on to other indexing techniques.

Need For Indexing: Speed

Consider the "Find" operation in Windows; a user search is initiated and a search starts through each file on the hard disk. When a directory is encountered, the search continues through each directory. With only a few thousand files on a typical laptop, a typical "find" operation takes a minute or longer.

Assume we have a fast processor executing an efficient sequential search algorithm. Further, assume we have a high speed disk drives that enable a scan of one million files per second (assume a document to be 5KB). This implies a scan rate of 5 GB per second, a very optimistic assumption. Currently, the world's largest web search engine indexes at least eight billion documents. This roughly puts the search time at around 1,600 seconds or roughly 26 minutes, and this is for a single user request! No visitor is going to wait for 26 minutes (not even 26 seconds) for a response. Hence, a sequential scan is simply not feasible.

Need For Indexing: Query Complexity

- How many customers do I have in Karachi?
- How many customers in Karachi made calls during April?
- How many customers in Karachi made calls to Multan during April?
- What are the average call charges per customer in Karachi in April?

Consider a typical sales database. It will consist of a number of tables. For instance, the customer table will consist of many rows of data, where each row is a unique customer. The first field in each row might be the customer ID number, followed by fields identifying first name, last name, address (that's actually several fields), phone number, and so forth. Another table might identify products with a unique product ID and fields identifying price, brand, inventory level, etc.

Now, imagine a simple query: how many customers are there in Karachi? The simplest way to do this is called a table scan: merely read every row sequentially and count the number of times you find a CITY field containing KHI. In a large database, of course, this will take a very long time. In this case, a simple index will greatly speed up data access by indexing the customer table by the city field. Of course, this means that a Database Administrator (DBA) has to create the index and keep it up to date.

Now, consider a more complex query: how many customers in Karachi made calls during April ? Now, the database must perform what is called a table join: it must look at both the CALLS table, qualifying the CALL DATE field in terms of a date range, and the Customer table, looking at the CITY field. Table joins are also very slow. In this case, a DBA might tune the database by creating a summary table containing customer calls by month and, again, the DBA would have to keep it up to date.

An even more complex query would be to ask for the records of customers in Karachi that what are the average call charges per customer in Karachi in April? This is a multidimensional query that requires building a multi-part key for another index or one might call for an aggregated result that is it may require all of the above operations plus the mathematical averaging of the $AMOUNT fields from all the qualifying Calls. Again, a DBA can create and update a summary table to handle this kind of query.

Need For Indexing: I/O Bottleneck

Throwing more hardware at the problem doesn't really help, either. Expensive and multi-processing servers can certainly accelerate the CPU-intensive parts of the process, but the bottom line of database access is disk access, so the process is I/O bound and I/O doesn't scale as fast as CPU power. You can get around this by putting the entire database into main memory, but the cost of RAM for a multi-gigabyte database is likely to be higher than the server itself! Therefore we index.

Although DBAs can overcome any given set of query problems by tuning, creating indexes, summary tables, and multiple data marts, or forbidding certain kinds of queries, they must know in advance what queries users want to make and would be useful, which requires domain-specific knowledge they often don't have. While 80% of database queries are repetitive and can be optimized, 80% of the ROI fro m database information comes from the 20% of queries that are not repetitive. The result is a loss of business or competitive advantage because of the inability to access the data in corporate databases in a timely fashion.

Indexing Concept

Indexing is purely a physical database concept and has nothing to do with the logical model. In fact an index should be completely invisible to someone who is doing the programming on the database. You should never access the index directly, it is the optimizer that should chose to use the index when it is appropriate to do so. The reality is of course is that the programmer generally will know what are the performance implications? But these implications should never affect the answer that I will get. Should only affect how long it takes to get the answer.

Think about an index as an analogy with a library. Think of the library as a collection of books, and there is huge room full of books. If you are looking for a particular book, unless the books are sorted by the title of the book, the one way you can look for that book is that you start out with the left hand wall and then you start looking at each and every book until you find the book you are interested in or you don't. So on average if there are *n* books in the library how long dose it take? It will take *O (n)* or *n/2* (on average). So it is an *O (n)* algorithm for accessing the data.

Now consider using the card catalog. The card catalog is organized in many different ways. By author, by topic by title, by what ever you want. Each of these have a different index. I have index on author, I have index on title and I have index on topic etc. so it takes me a little bit of extra time to go to the catalog first, but the catalog is sorted, so I can very quickly find the author that I am looking for because it is sorted and then I look at the card and know exactly which shelf to find the book. The catalog has some numbering system, which acts as a pointer, and points to shelf and the row on the shelf where the book is located. So I have to do a little bit of extra work to go to the catalog but then it takes me directly to the book I am interested in, that is what Indexing is. Indexing is just a card catalog to get access to the data i.e. efficient access to data. There is no actual information that I really want to get from the card catalog about the book. There is no data it is just an efficient way of accessing the book that I want. So the index in a database is just like that.

Indexing Goal

Look at as few blocks as possible to find the matching record(s)

The point of using an index is to increase the speed and efficiency of searches of the database. Without some sort of index, a user's query must sequentially scan the database, finding the records matching the parameters in the WHERE clause.

Conventional indexes

- Basic Types:
  - Sparse
  - Dense
  - Multi-level (or B-Tree)
- Primary Index vs. Secondary Indexes

Dense Index: Concept

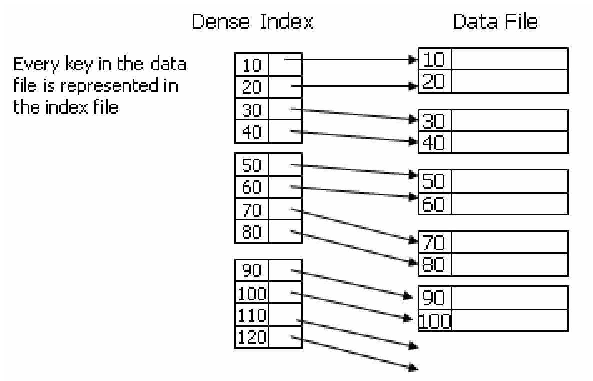


Figure-26.1: Dense index concept

Dense Index: Adv. & Dis. Adv.

For each record store the key and a pointer to the record in the sequential file. Why?

It uses less space, hence less time to search. Time (I/Os) logarithmic in number of blocks used by the index. Can also be used as secondary index, i.e. with another order of records.

Dense Index: Every key in the data file is represented in the index file

Pro:
A dense index, if fits in the memory, costs only one disk I/O access to locate a record given a key

Con:
A dense index, if too big and doesn't fit into the memory, will be expense when used to find a record given its key

Sparse Index: Concept

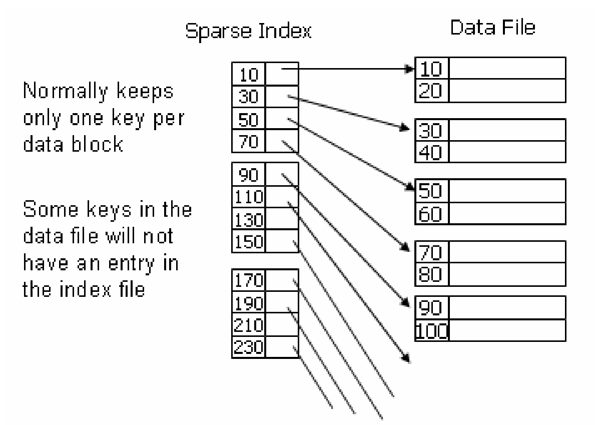

Sparse Index
Data File
Normally keeps only one key per data block
Some keys in the data file will not have an entry in the index file
10
30
50
70
90
110
130
150
170
190
210
230
10
20
30
40
50
60
70
80
90
100

Figure-26.2: Sparse index concept

In this case, normally only one key per data block is kept. A sparse index uses less space at the expense of somewhat more time to find a record given its key.
What happens when record 35 is inserted?

Sparse Index: Adv & Dis Adv

• Store first value in each block in the sequential file and a pointer to the block.

• Uses even less space than dense index, but the block has to be searched, even for unsuccessful searches.

• Time (I/Os) logarithmic in the number of blocks used by the index.

Sparse Index: Multi level

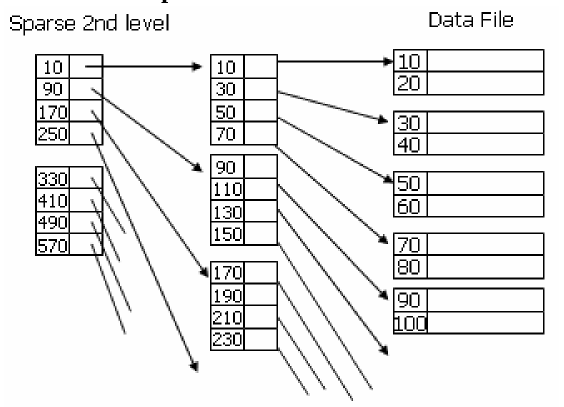


Figure-26.3: Sparse Index: Multi level

B-tree Indexing

- Can be seen as a general form of multi-level indexes.
- Generalize usual (binary) search trees (BST).
- Allow efficient and fast exploration at the expense of using slightly more space.
- Popular variant: B+-tree
- Support more efficiently queries like:

  SELECT * FROM R WHERE a = 11
  SELECT * FROM R WHERE 0<= b and b<42

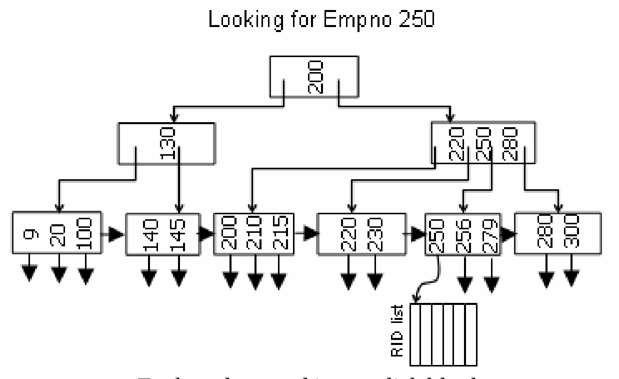
B-tree indexes are the most common index type used in typical OLTP applications and provide excellent levels of functionality and performance. Used in both OLTP and data warehouse applications, they speed access to table data when users execute queries with varying criteria, such as equality conditions and range conditions. B-tree indexes improve the performance of queries that select a small percentage of rows from a table. As a general guideline, an index should be created on tables that are often queried for less than a few percent (i.e. between two and four is a general guideline). Note that this is specific to a vendor.

B-tree indexes are stored in a tree structure that has branch blocks which point to lower-level blocks. The lowest level index blocks are called leaf blocks, and these blocks contain every indexed data value and a corresponding ROWID. The ROWID is a pointer that points directly back to the row in the data table.

When an index is accessed and the data needed to satisfy the query resides in the index leaf block, then there is no need to follow the ROWID pointer back to the table. In this case, the index can serve as the source of data, rather than the table.

In some applications, inserts of sequential keys results in a disk 'hot spot' at the leaf blocks. In this case, a B-tree index can be created with an option to improve performance and reduce I/O contention of leaf blocks by creating the index as a reverse-key index.

B-tree Indexing: Example



Each node stored in one disk block

Figure-26.4: B-tree Indexing: Example

Traditional relational database indexing is fairly uniform across products and generally comes in two flavors: B-trees and hashes, each of which has its strengths and weaknesses. For the most part, neither one is satisfactory for decision-support environments.

1. These indexes essentially map column values of rows in a single table and provide no information about the intersection (join) of these values across tables.

2. The content of the index is either the attribute value; a unique, internal identifier of each row, called a row ID or RID; or the unique key of the row that contains it. When tables reach tens or hundreds of millions of rows, these indexes become unwieldy, consuming huge amounts of memory and/or causing swapping and thrashing in memory.

3. The index structures are organized for transaction databases and do not take advantage of the stability of the data in a data warehouse between bulk update periods.

B-tree Indexing: Limitations

- If a table is large and there are fewer unique values.

  Capitalization is not programmatically enforced (meaning case-sensitivity does matter and "FLASHMAN" is different from "Flashman").

- Outcome varies with inter-character spaces.
- A noun spelled differently will result in different results.
- Insertion can be very expensive.

There are certain instances when a B-tree index is not appropriate and will not improve performance of queries. In many of these instances, such as a column in a data warehouse with relatively few distinct values, a bitmapped index can be created to dramatically improve performance. B-tree index is a poor choice for name and text searches because it is case-sensitive and requires a left-to-right match.

B-tree Indexing: Limitations Example

Given that MOHAMMED is the most common first name in Pakistan, a 5-million row Customers table would produce many screens of matching rows for MOHAMMED AHMAD, yet would skip potential matching values such as the following:

VALUE MISSED	REASON MISSED
Mohammed Ahmad	Case sensitive
MOHAMMED AHMED	AHMED versus AHMAD
MOHAMMED AHMAD	Extra space between names
MOHAMMED AHMAD DR	DR after AHMAD
MOHAMMAD AHMAD	Alternative spelling of MOHAMMAD

Table-26.1: B -tree Indexing: Limitations Example

For example, a B-tree search such as the following might be performed to find a customer:

SELECT * FROM Customers WHERE CustName="MOHAMMED AHMAD"

Given that MOHAMMED is the most common first name in Pakistan, a 5-million row Customers table would produce many screens of matching rows, yet would skip potential matching values as shown in Table 26.1.

Even when B-tree indexes are applied to columns they work well on, only one B-tree index can be used at a time. So, even if there are multiple indexes on one table, they cannot be used in combination. One index is picked by the Optimizer, and the selected rows are retrieved and then scanned for the other criteria.

Hash Based Indexing

- You may recall that in internal memory, hashing can be used to quickly locate a specific key.
- The same technique can be used on external memory.
- However, advantage over search trees is smaller in external search than internal. **WHY?**

Because part of search tree can be brought into the main memory.

Hash Based Indexing: Concept

In contrast to B-tree indexing, hash based indexes do not (typically) keep index values in sorted order.

- Index entry is found by hashing on index value requiring exact match.

  SELECT * FROM Customers WHERE AccttNo= 110240

- Index entries kept in hash organized tables rather than B-tree structures.
- Index entry contains ROWID values for each row corresponding to the index value.
- Remember few numbers in real-life to be useful for hashing.

A hash scan requires an exact match of a key value. It's extremely fast and efficient as long as the exact value (typically a number) is known. It's best for things like account number, NID number, or part vehicle registration number.

For example, if a customer wanted to buy a car on an installment plan from a bank, the bank would typically need his/her exact account number, for instance 110240. The SQL syntax to look it up would be similar to the following:

SELECT * FROM Customers WHERE AccttNo= 110240

A hash scan on account number, part number, order number or registration number is a fast, precise way of looking up information, but it's not the most customer- friendly or user-friendly. It requires knowledge of an arbitrary number that people often don't know without looking it up. How many of us actually know our bank account number from which we routinely draw money throughout the month?

For example, think of how many arbitrary numbers there are in your life all your customer account numbers, your phone number, address, your checking account number, savings account number(s), relatives' mobile numbers, all your credit card numbers, ATM PIN numbers, and so on. Imagine what it would take to memorize all of them.

Hashing as Primary Index

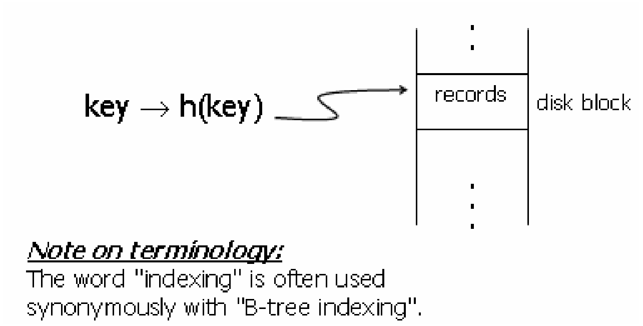

key → h(key)
records
disk block
Note on terminology:
The word "indexing" is often used synonymously with "B-tree indexing".

Figure-26.5: Hashing as Primary Index

Hashing as Secondary Index

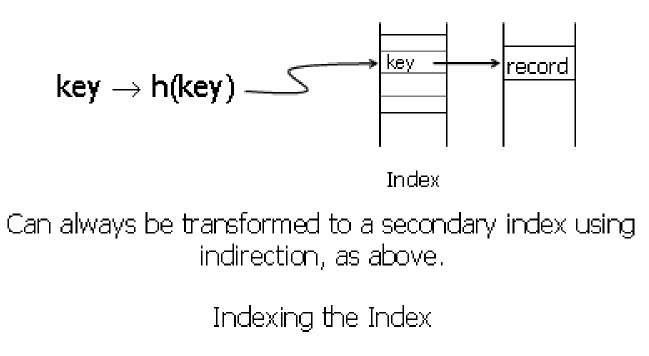

key → h(key)
key
record
Index
Can always be transformed to a secondary index using indirection, as above.
Indexing the Index

Figure-26.6: Hashing as Secondary Index

B-tree vs. Hash Indexes

- Indexing (using B-trees) good for range searches, e.g.:

  SELECT * FROM R WHERE A > 5

- Hashing good for match based searches, e.g.:

  SELECT * FROM R WHERE A = 5

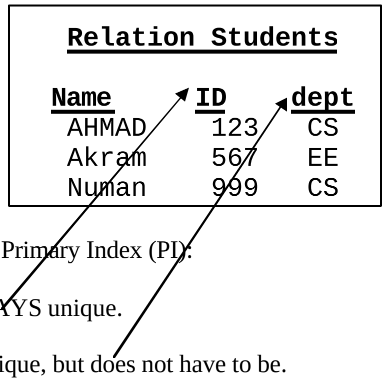
A B-tree index is more functional and flexible than a hash scan. It allows partial values to be specified as the retrieval criteria, and it brings back the rows in sorted order. It is most useful for fixed-structure or hierarchical columns such as dates or financial account numbers. Hash based indexing is fast for specific value searches as it takes *O(1)* for uniform hashing as opposed to *O(log n)* time for a B-Tree based index.

For example, a B-tree index on a Billing Date column would quickly find rows for the following partial date for customers on an installment plan:

SELECT * FROM Customers WHERE BillingDate="8502*"

A B-tree index has been used in many cases to provide access by name rather than just number. The typical customers know their name better than their customer numbers! So, historically, a B-tree index on a column such as Company Name or Contact Name has been far better than no name lookup at all.

Primary Key vs. Primary Index



Relation Students

Name	ID	dept
AHMAD	123	CS
Akram	567	EE
Numan	999	CS

- Primary Key (PK) & Primary Index (PI):
    - PK is ALWAYS unique.
    - PI can be unique, but does not have to be.
- In DSS environment, very few queries are PK based.

Primary Indexing: Criterion

- Primary index selection criteria:
    - o Common join and retrieval key.
    - o Can be Unique UPI or Non-unique NUPI.
    - o Limits on NUPI.
    - o Only one primary index per table (for hash-based file system).

A typical DSS query may consist of a number of join operations and the queries have a low selectivity i.e. do not involve a PK. Therefore, to speed-up the query processing, the PI is formed on the common join column. For example *cust_ID* in the transaction table (which is not unique and *cust_ID* in customer table, which is unique).

When the number of entries increase for NUPI a very large number of collisions will occur for hash-based indexing. If chaining is performed to resolve collision, the number of entries (depending on the block size) can become so large that a page/block fault may occur resulting in an additional I/O.

For hash based file-system the table is partitioned across the partitions, hence there can only be one PI. However, in case of pure hash based indexing, there can be number of PI on the same table, as only the pointers to the RID will change.

Primary Indexing: Example

Call Table

call_id	decimal (15,0) NOT NULL
caller_no	decimal (10,0) NOT NULL
call_duration	decimal (15,2) NOT NULL
call_dt	date NOT NULL
called_no	decimal (15,0) NOT NULL

What should be the primary index of the call table for a large telecom company?

Table-26.2: Primary Indexing: Example

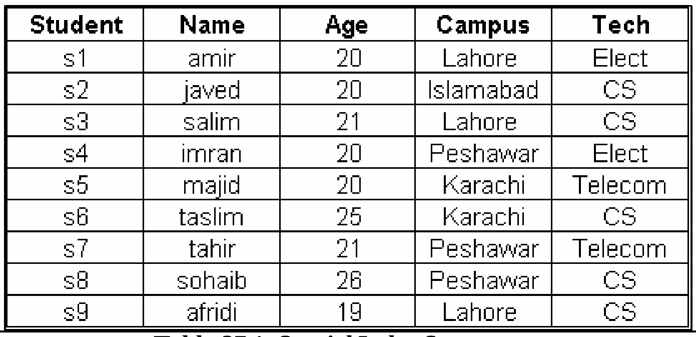
Lecture-27
Need for Speed: Special Indexing Techniques

Without indexes, the DBMS may be forced to conduct a full table scan (reading every row in the table) to locate the desired data, which can be a lengthy and inefficient process. However, creating indexes requires careful consideration. Although indexes can be quite useful for speeding data retrieval, they can slow performance of database writes. This slowdown occurs because a change to an indexed column actually requires two database writes -one to reflect a change in the table and one to reflect a corresponding change in the index. Thus, if the activities associated with a table are primarily write-intensive, it is important to make judicious use of indexes on the relevant tables. Indexes also require a certain amount of disk space, which must be considered when allocating resources to the database.

Before looking at your indexing options, we must first discuss the two ways to access data: non-keyed access and keyed access. Non-keyed access uses no index. Each record of the database is accessed sequentially, beginning with the first record, then second, third and so on. This access is good when you wish to access a large portion of the database (greater than 85%). Keyed access provides direct addressing of records. A unique number or character(s) is used to locate and access records. In this case, when specified records are required (say, record 120, 130, 200 and 500), indexing is much more efficient than reading all the records in between.

Special Index Structures

- Inverted index
- Bit map index
- Cluster index
- Join indexes



Student	Name	Age	Campus	Tech
s1	amir	20	Lahore	Elect
s2	javed	20	Islamabad	CS
s3	salim	21	Lahore	CS
s4	imran	20	Peshawar	Elect
s5	majid	20	Karachi	Telecom
s6	taslim	25	Karachi	CS
s7	tahir	21	Peshawar	Telecom
s8	sohaib	26	Peshawar	CS
s9	afridi	19	Lahore	CS

Table-27.1: Special Index Structures

Inverted index: Concept

An inverted index is an optimized structure that is built primarily for retrieval, with update being only a secondary consideration. The basic structure *inverts* the text so that instead of the view obtained from scanning documents where a document is found and then its terms are seen (think of a list of documents each pointing to a list of terms it contains), an index is built that maps terms to documents (pretty much like the index found in the back of a book that maps terms to page numbers). Instead of listing each *document* once (and each term repeated for each document that contains the term), an inverted index lists each *term* in the collection only once and then shows a list of all the documents that contain the given term. Each document identifier is repeated for each term that is found in the document.

Within the search engine domain, data are searched far more frequently than they are updated. This is typical for a data warehouse, where updates hardly take place. Given this situation a data structure called an *inverted index* commonly used by search engines is also applicable for the data warehouse environment.

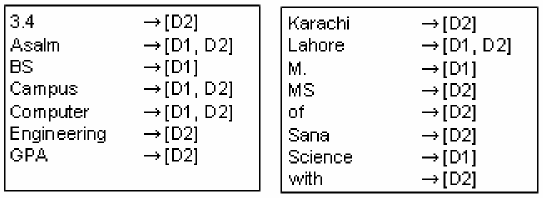
An inverted index is able to do many accesses in *O (1)* time at the price of significantly longer time to do an update, in the worst case O(*n*). Index construction time is longer as well, but query time is generally faster than with a B-tree i.e. O(log n). Since index construction is an off-line activity, so it is an appropriate tradeoff i.e. shorter query times at the expense of lengthier index construction times.

Finally, inverted index storage structures can exceed the storage demands of the document collection itself. However, the inverted index can be compressed for many systems, to around 10% of the original document collection. Given the alternative (of 26 minute searches), search engine developers are happy to trade index construction time and storage for query efficiency. Same is also true for the DSS environment.

Inverted Index: Example-1

D1: M. Asalm BS Computer Science Lahore Campus
D2: Sana Aslam of Lahore MS Computer Engineering with GPA 3.4 Karachi Campus

Inverted index for the documents D1 and D2 is as follows:



3.4	→ [D2]
Asalm	→ [D1, D2]
BS	→ [D1]
Campus	→ [D1, D2]
Computer	→ [D1, D2]
Engineering	→ [D2]
GPA	→ [D2]

Karachi	→ [D2]
Lahore	→ [D1, D2]
M.	→ [D1]
MS	→ [D2]
of	→ [D2]
Sana	→ [D2]
Science	→ [D1]
with	→ [D2]

Inverted Index: Example-2

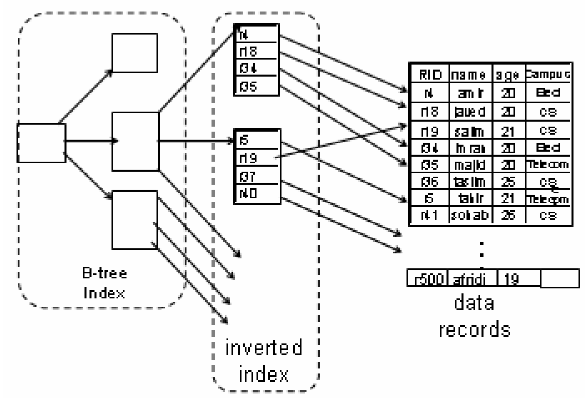


Figure-27.1: Inverted Index: Example-2

Inverted Index: Query

- Query:
  - Get students with age = 20 and tech = “telecom”
- List for age = 20: r4, r18, r34, r35
- List for tech = “telecom”: r5, r35
- Answer is intersection: r35

Bitmap Indexes: Concept

- Index on a particular column
- Index consists of a number of bit vectors or bitmaps
- Each value in the indexed column has a corresponding bit vector (bitmaps)
- The length of the bit vector is the number of records in the base table
- The *i*th bit is set to 1 if the *i*th row of the base table has the value for the indexed column

Bitmap Indexes: Example

- The index consists of bitmaps, with a column for each unique value:

Index on City (larger table):

SID	Islamabad	Lahore	Karachi	Peshawar
1	0	1	0	0
2	1	0	0	0
3	0	1	0	0
4	0	0	0	1
5	0	0	1	0
6	0	0	1	0
7	0	0	0	1
8	0	0	0	1
9	0	1	0	0

Index on Tech (smaller table):

SID	CS	Elect	Telecom
1	1	0	0
2	0	1	0
3	0	1	0
4	1	0	0
5	0	0	1
6	0	1	0
7	0	0	1
8	1	0	0
9	1	0	0

Table-27.2: Bitmap Indexes: Example

Bitmap Index: Query

- Query:
  - Get students with age = 20 and campus = “Lahore”
- List for age = 20: 1101100000
- List for campus = “Lahore”: 1010000001
- Answer is AND : 1000000000
- Good if domain cardinality is small
- Bit vectors can be compressed
  - Run length encoding

Bitmap Index: Run Length Encoding

Basic Concept

Case-1	1111000011110000001111100000011111 INPUT 14#04#14#06#15#06#15 OUTPUT
Case-2	1010101010101010101010101010101010 INPUT 11#01#11#01#11#01#11#01#… OUTPUT
Case-3	1111111111111111110000000000000000 INPUT 117#017 OUTPUT

Bitmap Index: More Queries

- Allow the use of efficient bit operations to answer queries, such as:
  - "Which students from Lahore are enrolled in 'CS'?"
    Perform a bit-wise AND of two bitmaps: answer – s1 and s9
  - "How many students are enrolled in 'CS'?"
    Count 1's in the degree bitmap vector

Answer is 4

Bitmaps are not good for high- cardinality textual columns that have numerous values, such as names or descriptions, because a new column is created in the index for every possible value. With high-cardinality data and a large number of rows, each bitmap index becomes enormous and takes a long time to process durin g indexing and retrievals.

Bitmap Index: Adv.

- Very low storage space.
- Reduction in I/O, just using index.
- Counts & Joins
- Low level bit operations.

An obvious advantage of this technique is the potential for dramatic reductions in storage overhead. Consider a table with a million rows and four distinct values with column header of 4 bytes resulting in 4 MB. A bitmap indicating which of these rows are for these values requires about 500KB.

More importantly, the reduction in the size of index "entries" means that the index can sometimes be processed with no I/O and, more often, with substantially less I/O than would otherwise be required. In addition, many index-only queries (queries whose responses are derivable through index scans without searching t he database) can benefit considerably.

Database retrievals using a bitmap index can be more flexible and powerful than a B-tree in that a bitmap can quickly obtain a count by inspecting only the index, without retrieving the actual data. Bitmap indexing can also use multiple columns in combination for a given retrieval.

Finally, you can use low-level Boolean logic operations at the bit level to perform predicate evaluation at increased machine speeds. Of course, the combination of these factors can result in better query performance.

Bitmap Index: Performance Guidance

Bitmapped indexes can provide very impressive performance speedups; execution times of certain queries may improve by several orders of magnitude. The queries that benefit the most from bitmapped indexes have the following characteristics:

- The WHERE-clause contains multiple tests on low-cardinality columns
- The individual tests on these low-cardinality columns select a large number of rows
- The bitmapped indexes have been created on some or all of these low-cardinality columns
- The tables being queried contain many rows

A significant advantage of bitmapped indexes is that multiple bitmapped indexes can be used to evaluate the conditions on a single table. Thus, bitmapped indexes are very appropriate for complex ad-hoc queries that contain lengthy WHERE-clauses. Performance, storage requirements, and maintainability should be considered when evaluating an indexing scheme.

Bitmap Index: Dis. Adv.

- Locking of many rows
- Low cardinality
- Keyword parsing
- Difficult to maintain - need reorganization when relation sizes change (new bitmaps)

Row locking: A potential drawback of bitmaps involves locking. Because a page in a bitmap contains references to so many rows, changes to a single row inhibit concurrent access for all other referenced rows in the index on that page.

Low cardinality: Bitmap indexes create tables that contain a cell for each row times each possible value (the product of the number of rows times the number of unique values). Therefore, a bitmap is practical only for low- cardinality columns that divide the data into a small number of categories, such as "M/F", "T/F", or "Y/N" values.

Keyword parsing: Bitmap indexes can parse multiple values in a column into separate keywords. For example, the title “Marry had a little lamb" could be retrieved by entering the word “Marry" or “lamb" or a combination. Although this keyword parsing and lookup capability is extremely useful, textual fields tend to contain high-cardinality data (a large number of values) and therefore are not a good choice for bitmap indexes.

Cluster Index: Concept

- A Cluster index defines the sort order on the base table.
- Ordering may be strictly enforced (guaranteed) or opportunistically maintained.
- At most one cluster index defined per table.
- Cluster index may include one or multiple columns.
- Reduced I/O.

The big advantage of a cluster index is that all the rows with the same cluster index value will be placed into adjacent locations in a small number of data blocks.

In this example, all accounts for a specific customer will be clustered into adjacent locations to increase locality of reference by *customer_id*.

Cluster indexing allows significant reduction in I/Os when accessing base table via index because rows with the same index value will be stored into the same blocks. This is a big win for indexed access for query execution against a single table as well as nested loop joins using indexed access. Cluster indexing has the effect of transforming a random I/O workload into a sequential I/O workload when accessing through the cluster index.

Cluster Index: Example

Student	Name	Age	Campus	Tech
s9	afridi	19	Lahore	CS
s1	amir	20	Lahore	Elect
s2	javed	20	Islamabad	CS
s4	imran	20	Peshawar	Elect
s5	majid	20	Karachi	Telecom
s3	salim	21	Lahore	CS
s7	tahir	21	Peshawar	Telecom
s6	taslim	25	Karachi	CS
s8	sohaib	26	Peshawar	CS

Cluster indexing on AGE

One indexing column at a time

Student	Name	Age	Campus	Tech
s9	afridi	19	Lahore	CS
s2	javed	20	Islamabad	CS
s3	salim	21	Lahore	CS
s6	taslim	25	Karachi	CS
s8	sohaib	26	Peshawar	CS
s1	amir	20	Lahore	Elect
s4	imran	20	Peshawar	Elect
s5	majid	20	Karachi	Telecom
s7	tahir	21	Peshawar	Telecom

Cluster indexing on TECH

Table-27.3: Cluster Index: Example

Cluster Index: Issues

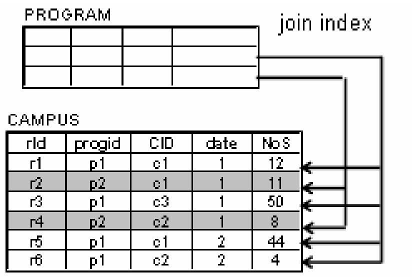
- Works well when a single index can be used for the majority of table accesses.
- Selectivity requirements for making use of a cluster index are much less stringent than for a non-clustered index.
    - Typically by an order of magnitude, depending on row width.
- High maintenance cost to keep sorted order or frequent reorganizations to recover clustering factor.
- Optimizer must keep track of clustering factor (which can degrade over time) to determine optimal execution plan

Significant performance advantage for query execution, but beware of the overhead of index maintenance (and reorganization costs!), at best *O (n log n).*

Query plans will change (or should change) over time where updates are occurring because degradation of clustering (e.g., adjacency) will take place over time until a reorganization is performed. Note that this effect assumes that clustering index does not "force" sorted order, but rather that sorted order is achieved with initial index build and periodic reorganization. It is also possible to "force" sorted order at a higher cost upon insert and update operations into the data warehouse.

Join Index: Example

The rows of the table consist entirely of such references, which are the RIDs of the relevant rows.



rId	progid	CID	date	NoS
r1	p1	c1	1	12
r2	p2	c1	1	11
r3	p1	c3	1	50
r4	p2	c2	1	8
r5	p1	c1	2	44
r6	p1	c2	2	4

Figure-27.2: Join Index: Example

Lecture-28
Join Techniques

Background

Used to retrieve data from multiple tables.

Joins used frequently, hence lot of work on improving or optimizing them.

Simplest join that works in most cases is nested-loop join but results in quadratic time complexity.

Tables identified by FROM clause and condition by WHERE clause.

Will cover different types of joins.

Join commands are statements that retrieve data from multiple tables. A join is identified by multiple tables in the `FROM` clause, and the relationship between the tables to be joined is established through the existence of a join condition in the `WHERE` clause. Because joins are so frequently used in relational queries and because joins are so expensive, lot of effort has gone into developing efficient join algorithms and techniques. The simplest i.e. nested-loop join is applicable in all cases, but results in quadratic performance. Several fast join algorithms have been developed and extensively used; these can be categorized as sort-merge, hash-based, and index-based algorithms. In this lecture we will be covering the following join algorithms/techniques:

- Nested loop join
- Sort Merge Join
- Hash based join
- Etc.

About Nested-Loop Join

Typically used in OLTP environment.

Limited application for DSS and VLDB

In DSS environment we deal with VLDB and large sets of data.

Traditionally Nested-Loop join has been and is used in OLTP environments, but for many reasons, such a join mechanism is not suitable for VLDB and DSS environments. Nested loop joins are useful when small subsets of data are joined and if the join condition is an efficient way of accessing the inner table. Despite these restrictions/limitations, we will begin our discussion with the traditional join technique i.e. nested loop join, so that you can appreciate the benefits of the join techniques typically used in a VLDB and DSS environment.

Nested-Loop Join: Code

```
FOR  i = 1 to n DO BEGIN        /*         N rows in T1                 */
          IF ith row of T1 qualifies THEN BEGIN
                    For j = 1 to m DO BEGIN         /*       M rows in T2                  */
                      IF the ith row of T1 matches to jth row of T2 on join key THEN BEGIN
                        IF the jth row of T2 qualifies THEN BEGIN
                           produce output row
                        END
            END
          END
      END
 END
```

Nested loop join works like a nested loop used in a high level programming language, where each value of the index of the outer loop is taken as a limit (or starting point or whatever applicable) for the index of the inner loop, and corresponding actions are performed on the statement(s) following the inner loop. So basically, if the outer loop executes R times and for each such execution the inner loop executes S times, then the total cost or time complexity of the nested loop is *O(RS)*.

Nested-loop joins provide efficient access when tables are indexed on join columns. Furthermore, in many small transactions, such as those affecting only a small set of rows, index nested loops joins are far superior to both sort-merge joins and hash joins. A nested loop join involves the following steps:

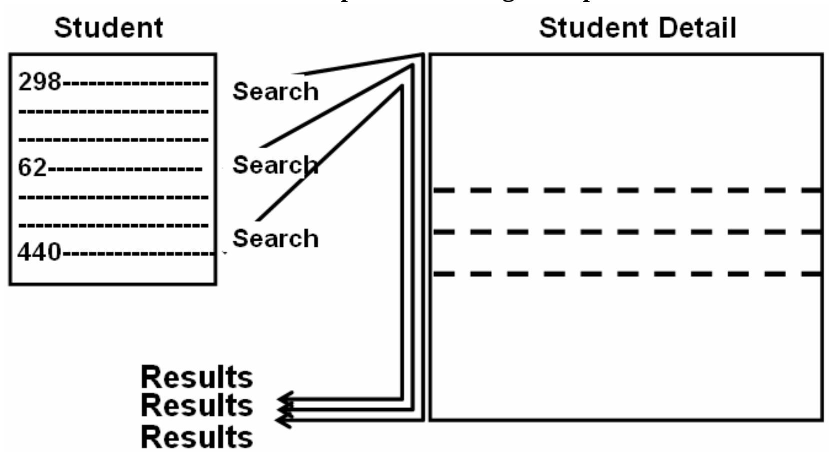
1. The optimizer determines the major table (i.e. *Table_A*) and designates it as the outer table. *Table_A* is accessed once. If the outer table has no useful indexes, a full table scan is performed. If an index can reduce I/O costs, the index is used to locate the rows.

2. The other table is designated as the inner table or *Table_B*. *Table_B* is accessed once for each qualifying row (or touple) in *Table_A*.

3. For every row in the outer table, DBMS accesses all the rows in the inner table. The outer loop is for every row in outer table and the inner loop is for every row in the inner table.

If 10 rows from *Table_A* match the conditions in the query, *Table_B* is accessed 10 times. If *Table_B* has a useful index on the join column, it might require 5 I/Os to read the data block for each scan, plus one I/O for each data block. Hence the total cost of accessing *Table_B* would be 60 logical I/Os.

Nested-Loop Join: Working Example



For each qualifying row of Personal table,
Academic table is examined for matching rows.

Figure-28.1: The process of Nested-Loop Join without indexing

The process of creating the result set for a nested-loop join is achieved by nesting the tables, and scanning the inner table repeatedly for each qualifying row of the outer table. This process is shown in Figure-28.1: The question being asked is ***"What is the average GPS of undergraduate male students?"***.

Nested-Loop Join: Order of Tables

If the outer loop executes R times and the inner loop executes S times, then the time complexity will be O(RS).

The time complexity should be independent of the order of tables i.e. O(RS) is same as O(SR).

However, in the context of I/Os the order of tables does matter.

Along with this the relationship between the number of qualifying rows/blocks between the two tables matters.

It is true that the actual number of matching rows returned as a result of the join would be independent of the order of tables i.e. inner or outer; actually it is going to be even independent of the type of join used. Now assume that depending on the filter criterion, different number of rows is returned from the two tables to be joined. Let the rows returned from Table_A be R_A and rows returned from Table_B be R_B, and let $R_A < R_B$. If Table_B would have been the outer table, then for each row returned, the inner table i.e. Table_A would have been scanned (assuming no indexing) R_B times. However, if Table_A would have been the outer table, then for each row returned, the inner table i.e. Table_B would have been scanned (assuming no indexing) R_A times i.e. less number of times scanned. Meaning, even if the inner table have fewer number of qualifying rows, it is going to be scanned the number of times the qualifying rows of the outer table. Hence the choice of tables does matter.

Nested-Loop Join: Cost Formula

Join cost = Cost of accessing *Table_A* +
# of qualifying rows in *Table_A* × Blocks of *Table_B* to be scanned for each qualifying row

OR

Join cost = Blocks accessed for *Table_A* +
Blocks accessed for *Table_A* × Blocks accessed for *Table_B*

For a high level programming language, the time complexity of a nested-loop join remains unchanged if the order of the loops are changed i.e. the inner and outer loops are interchanged. However, this is NOT true for Nested-Loop-Joins in the context of DSS when we look at the I/O. For a nested-loop join with two tables, the formula for estimating the join cost is:

Join cost = Cost of accessing *Table_A* +
# of qualifying rows in *Table_A* × Blocks of *Table_B* to be scanned for each qualifying row.

OR

Join cost = Blocks accessed for *Table_A* +
Blocks accessed for *Table_A* × Blocks accessed for *Table_B*

For a Nested-Loop join inner and outer tables are determined as follows:

The outer table is usually the one that has:

- The smallest number of qualifying rows, and/or
- The largest numbers of I/Os required to locate the rows.

The inner table usually has:

- The largest number of qualifying rows, and/or

The smallest number of reads required to locate rows.

Nested-Loop Join: Cost of reorder

Table_A = 500 blocks and
Table_B = 700 blocks.

Qualifying blocks for *Table_A QB(A)* = 50
Qualifying blocks for *Table_B QB(B)* = 100

Join cost A&B = 500 + 50×700 = 35,500 I/Os
Join cost B&A = 700 + 100×500 = 50,700 I/Os

i.e. an increase in I/O of about 43%.

For example, if qualifying blocks for *Table_A* QB(A) = 50 and qualifying blocks for *Table_B* QB(B) = 100 and size of Table_A is 500 blocks and size of Table_B is 700 blocks then Join cost A&B = 500 + 50×700 = 35,500 I/Os and using the other order i.e. *Table_B* outer table and *Table_A* as inner table, the join cost B&A = 700 + 100×500 = 50,700 I/Os i.e. an increase in I/O of about 43%.

Nested-Loop Join: Variants

1. Naive nested-loop join
2. Index nested-loop join
3. Temporary index nested-loop join

Working of Query optimizer

There are many variants of the traditional nested-loop join. The simplest case is when an entire table is scanned; this is called a naive nested-loop join. If there is an index, and that index is exploited, then it is called an index nested-loop join. If the index is built as part of the query plan and subsequently dropped, it is called as a temporary index nested-loop join. All these variants are considered by the query optimizer before selecting the most appropriate join algorithm/technique.

Sort-Merge Join

Joined tables to be sorted as per WHERE clause of the join predicate.

Query optimizer scans for (cluster) index, if exists performs join.

In the absence of index, tables are sorted on the columns as per WHERE clause.

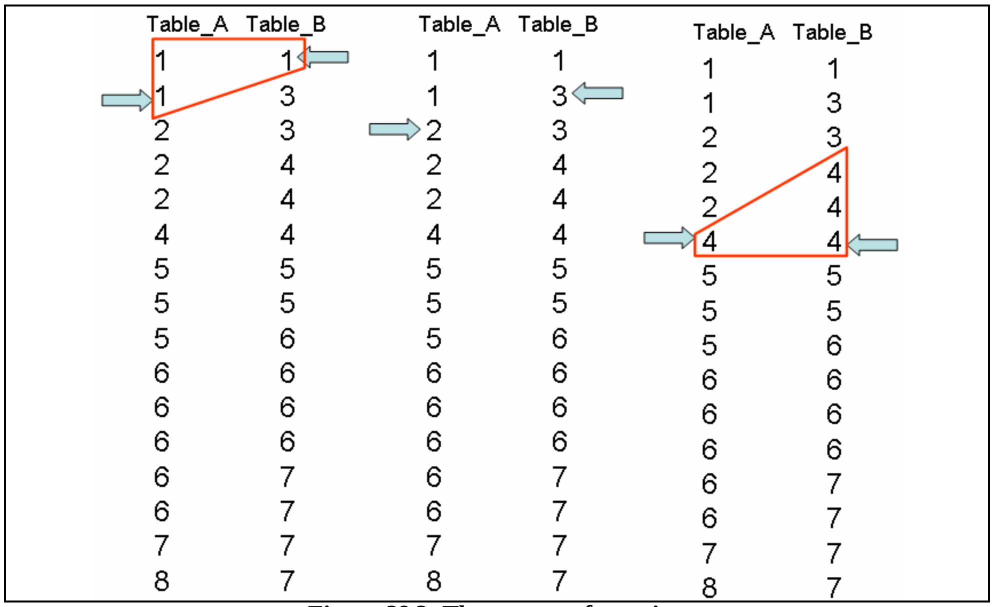
If multiple equalities in WHERE clause, some merge columns used.

The Sort-Merge join requires that both tables to be joined are sorted on those columns that are identified by the equality in the `WHERE` clause of the join predicate. Subsequently the tables are merged based on the join columns. The query optimizer typically scans an index on the columns which are part of the join, if one exists on the proper set of columns, fine, else the tables are sorted on the columns to be joined, resulting in what is called a cluster index. However, in rare cases, there may be multiple equalities in the `WHERE` clause, in such a case, the merge columns are taken from only some of the given equality clauses.

Because each table is sorted, the Sort-Merge Join operator gets a row from each table and compares it one at a time with the rows of the other table. For example, for equi-join operations, the rows are returned if they match/equal on the join predicate. If they are not equal or don't match, whic hever row has the lower value is discarded, and next row is obtained from that table. This process is repeated until all the rows have been exhausted.

The Sort-Merge join process just described works as follows:

- Sort *Table_A* and *Table_B* on the join column in ascending order, then scan them to do a ``merge'' (on join column), and output result tuples/rows.
    - Proceed with scanning of *Table_A* until current A_tuple ≤ current B_tuple, then proceed scanning of *Table_B* until current B_tuple ≤ current A_tuple; do this until current A_tuple = current B_tuple.
    - At this point, all A_tuples with same value in Ai (current A_group) and all B_tuples with same value in Bj (current B_group) match; output <a, b> for all pairs of such tuples/records.
    - Update pointers, resume scanning *Table_A* and *Table_B*.
- *Table_A* is scanned once; each B group is scanned once per matching *Table_A* tuple. (Multiple scans of a B group are likely to find needed pages in buffer.)
- Cost: M log M + N log N + (M+N)
    - The cost of scanning is M+N, could be M*N (very unlikely!)



Table_A	Table_B	Table_A	Table_B	Table_A	Table_B
1	1	1	1	1	1
1	3	1	3	1	3
2	3	2	3	2	3
2	4	2	4	2	4
2	4	2	4	2	4
4	4	4	4	4	4
5	5	5	5	5	5
5	5	5	5	5	5
5	6	5	6	5	6
6	6	6	6	6	6
6	6	6	6	6	6
6	6	6	6	6	6
6	7	6	7	6	7
6	7	6	7	6	7
7	7	7	7	7	7
8	7	8	7	8	7

Figure-28.2: The process of merging

Fig-28.2 shows the process of merging two sorted tables with IDs shown. Conceptually the merging is similar to the merging you must have studies in Merge_Sort in your Algorithm course.

Sort-Merge Join: Note

Very fast.

Sorting can be expensive.

Presorted data can be obtained from existing B-tree.

Sort-Merge join itself is very fast, but it can be an expensive choice if sort operations are required frequently i.e. the contents of the table's change often resulting in deterioration of the sort order. However, it may so happen that even if the data volume is large the desired data can be obtained presorted from existing B-tree. For such a case sort -merge join is often the fastest available join algorithm.

Hash-Based Join: Working

Suitable for the VLDB environment.

The choice which table first gets hashed plays a pivotal role in the overall performance of the join operation, this decided by the optimizer.

The joined rows are identified by collisions i.e. collisions are "good" in case of hash join.

Hash joins are suitable for the VLDB environment as they are useful for joining large data sets or tables. The choice about which table first gets hashed plays a pivotal role in the overall performance of the join operation, and left to the optimizer. The optimizer decides by using the smaller of the two tables (say) *Table_A* or data sources to build a hash table in the main memory on the join key used in the `WHERE` clause. It then scans the larger table (say) *Table_B* and probes the hashed table to find the joined rows. The joined rows are identified by collisions i.e. collisions are "good" in case of hash join.

The optimizer uses a hash join to join two tables if they are joined using an equijoin and if either of the following conditions are true:

- A large amount of data needs to be joined.
- A large portion of the table needs to be joined.

This method is best used when the smaller table fits in the available main memory. The cost is then limited to a single read pass over the data for the two tables. Else the "smaller" table has to be partitioned which results in unnecessary delays and degradation of performance due to undesirable I/Os.

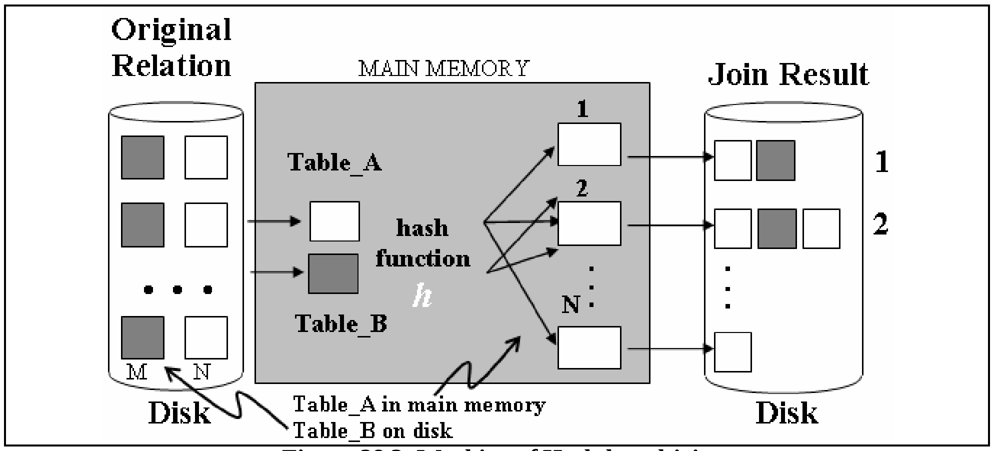


Figure-28.3: Working of Hash-based join

<u>Cost of Hash-Join</u>

- In partitioning phase, read + write both operations requires 2(M+N) I/Os.
- In matching phase, read both requires M+N I/Os.

Hash-Based Join: Large "small" Table

Smaller of the two tables may grow too big to fit into the main memory.

Optimizer performs partitioning, but is not simple.

Multi-step approach followed, each step has a build phase and probe phase.

Both tables entirely consumed and partitioned via hashing.

Hashing guarantees any two joining records will fall in same pair of partitions.

Task reduced to multiple, but smaller, instances of the same tasks.

It may so happen that the smaller of the two tables grows too big to fit into the main memory, then the optimizer breaks it up by partitioning, such that a partition can fit in the main memory. However, it is not that simple because the qualifying rows of both the tables have to fall in the corresponding partition pairs that are hashed (build) and probed. Thus in such a case the hash join proceeds in several steps. Each step has a build phase and probe phase. Initially, the two tables are entirely consumed and partitioned (using a hash function on the hash keys) into multiple partitions. The number of such partitions is sometimes called the partitioning fan-out. Using the hash function on the hash keys (based on the predicates in the `WHERE` clause) guarantees that any two joining records must be in the same pair of partitions. Therefore, the task of joining two large tables gets reduced to multiple, but smaller, instances of the same tasks. The hash join is then applied to each pair of partitions.

Hash-Based Join: Partition Skew

Partition skew can become a problem.

Hashing works under the assumption of uniformity of data distribution, may not be always true.

Consequently hash-based join degrades into nested-loop join.

Solution: Make available other hash functions to be chosen by the optimizer; that better distribute the input.

Partition skew can become a problem in hash-join. In the first step of hash join, records are hashed into the main memory into their corresponding bucket. This being done based on the hash function used. However, an attribute being hashed may not be uniformly distributed within the relation, and some buckets may then contain more records than other buckets. When this difference becomes large, the corresponding bucket may no longer fit in the main memory. As a consequence, hash-based join degrades into performance of a nested-loop join. The only possible solution is to make available other hash functions to be chosen by the optimizer; that better distribute the input.

Hash-Based Join: Intrinsic Skew

Intrinsic skew can become a problem for hash, as well as sort-merge join.

The skew is in data, NOT due to hash function.

Example: Many non-CS majors registering for CS-101 instead of CS students in summer.

Intrinsic skew occurs when attributes are not distributed uniformly; it is also called attribute value skew. For example a basic Computer Science (CS) course being offered in summer, and taken by many non-CS majors who want to know about computers. The course taken by few CS-majors who missed it or got an incomplete (i.e. I) grade during the regular semester due to one reason or another. Ironically, intrinsic skew effects the performance of both hash and sort-merge joins. Sort-merge join works best when the join attributes are the primary key of both tables. This property guarantees absence of duplicates, so that a record in the left-hand-side of the relation will join with at most one record in the right-hand-side of the relation, thus avoiding the intrinsic skew.

Lecture # 29
A Brief Introduction to Data mining (DM)

In our one of previous lectures, we discussed "putting the pieces together". One of the things in those pieces was data mining. We mine data when we need to discover something out of a lot. It is a broad discipline, dedicated courses being offered solely. However, we will go through a brief introduction of the field so that we get ample knowledge about data mining concepts, applications and techniques.

What is Data Mining?: Informal

"There are things that we know that we know…

there are things that we know that we don't know…

there are things that we don't know we don't know."

Donald Rumsfeld

US Secretary of Defence

Lets start data mining with a interesting statement. Why interesting because the statement covers the overall concept of DM and is given by a non-technical person who neither is a scientist nor a data mining guru. The statement, given by Donald Rumsfeld, Defense Secretary of the USA in an interview, is as under.

As we know, there are known knowns. There are things we know that we know like you know your names, your parent's names. We also know there are known unknowns. That is to say, we know that there are some things we do not know like what one is thinking about you, what you will eat after six days, what will be result of a lottery and so on. But there are also unknown unknowns, the ones we don't know that we don't know. Are they beneficial if you know? Or it is harmful no to know them?

There are also unknown knowns, things we'd like to know, but don't know, but know someone who can doctor them and pass them off as known knowns. To associate Rumsfeld's above quotation with data mining, we identify four core phrases as

1. Known knowns
2. Known unknowns
3. Unknown unknowns
4. Unknown knowns

The items 1 3, and 4 deal with "*Knowns*". Data mining has relevance to the third point in red. It is an art of digging out what exactly we don't know that we must know in our business. The methodology is to first convert "*unknown unkowns*" into "*known unknowns*" and then finally to *"known knowns"*.

What is Data Mining?: Slightly Informal

Tell me something that I should know.

When you don't know what you should be knowing, how do you write SQL?

You cant!!

Now a slightly technical view of DM. Not that much technical but you may easily understand. Tell me something that I should know i.e. you ask your DWH, data repository that tell me something that I don't know, or I should know. Since we don't know what we actually don't know and what we must know to know, we can't write SQL's for getting answers like we do in OLTP systems. Data mining is an exploratory approach, where browsing through data using data mining techniques may reveal something that might be of interest to the user as information that was unknown previously. Hence, in data mining we don't know the results.

What is Data Mining?: Formal

- Knowledge Discovery in Databases (KDD).
- Data mining digs out valuable non-trivial information from large multidimensional apparently unrelated data bases (sets).
- It's the integration of business knowledge, people, information, algorithms, statistics and computing technology.
- Finding useful hidden patterns and relationships in data.

Before looking into the technical or formal view of DM, consider the quote that you might have heard in your childhood i.e. *finding a needle in the haystack*. It is a tough job to find a needle in a big box full of hay. Dm is finding in the hay stack (huge data) the needle (knowledge). You don't have idea about where the needle can be found or even you don't know whether the needle is there in the haystack or not.

Historically, the notion of finding useful patterns in data has been given a variety of names, including data mining, knowledge extraction, information discovery, information harvesting, data archaeology, and data pattern processing. The term data mining has mostly been used by statisticians, data analysts, and the management information systems (MIS) communities. It has also gained popularity in the database field. KDD refers to the overall process of discovering useful knowledge from data, and data mining refers to a particular step in this process. Data mining is the application of specific algorithms for extracting patterns from data. The additional steps in the KDD process, such as data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, and proper interpretation of the results of mining, is essential to ensure that useful knowledge is derived from the data. Blind application of data-mining methods (rightly criticized as data dredging in the statistical literature) can be a dangerous activity, easily leading to the discovery of meaningless and invalid patterns.

An important to note here is the use of term *discovery* rather than *finding*. This is so because you find something that you know you have lost, or you know that something exist but not in your approach so you search for that and find. In DM as we are saying again and again you don't know what you are looking for. In other words you don’t know the results, so the term discovery rather than finding is used.

In the given definition, the 5 key words are;

Nontrivial

By *nontrivial*, we mean that some search or inference is involved; that is, it is not a straightforward computation of predefined quantities like computing the average value of a set of numbers. For example, suppose the majority of the customers of a garments shop are men. If some women too buy garments then it’s common and trivial because women sometimes buy for their children and spouse. Similarly if “Suwaiyan” sale increases during Eid days, the sugar, milk and date sales also in crease. The information again is trivial. If the sale of some items boosts up, even when no Eid was around, in some region of the country, this is non-trivial information

Value

The term *value* refers to the importance of discovered hidden patterns to the user in terms of its usability, validity, benefit and understandability. Data mining is a way to intelligently probing large databases to find exactly where the *value* resides.

Multidimensional

By multidimensional we mean a database designed as a multidimensional hypercube with one axis per dimension. In a flat or relational database, each field in a record represents a dimension. In a multidimensional database, a dimension is a set of similar entities; for example, a multidimensional sales database might include the dimensions Product, Time, and City. Data mining Multidimensional databases allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified.

Unrelated

Humans often lack the ability to comprehend and manage the immense amount of available and unrelated data. Data mining can help us take very small pieces of data that are seemingly unrelated i.e. no relationship exits, and determine whether they are correlated and can tell us anything that we need to know”.

Business Knowledge

The domain business processes must be known apriority before applying defaming techniques. Since data mining is an exploratory approach we can not know what we should know, until and unless we are not well aware of the activities involved in current business processes.

People

Human involvement in the data mining process is crucial in sense that value of patterns is well known to the user. Since data mining focuses on *“unknown unkowns”* , people factor plays a key role in directing data mining probe in a direction that ultimately ends in something that is previously unknown, novel, and above all of value. Thus, Data mining has proven to be a powerful tool capable of providing highly targeted information to support decision-making and forecasting for people like scientists, physicians, sociologists, the military and business etc.

Algorithms

Data mining consists of algorithms for extracting useful patterns from huge data. Their goal is to make prediction or/and give description. Prediction involves using some variables to predict unknown values (e.g. future values) of other variables while description focuses on finding interpretable patterns describing the data. These algorithms can sift through the data in search of frequently occurring patterns, can detect trends, produce generalizations about the data, etc. There are different categories of data algorithms based on their application, approach etc. Commonly used are classification algorithms, clustering algorithms, rule based algorithms, and artificial neural networks etc.

Statistics

Data Mining uses statistical algorithms to discover patterns and regularities (or "knowledge") in data. For example: classification and regression trees (CART, CHAID), rule induction (AQ, CN2), nearest neighbors, clustering methods, association rules, feature extraction, data visualization, etc.

Data mining is, in some ways, an extension of statistics, with a few artificial intelligence and machine learning twists thrown in. Like statistics, data mining is not a business solution, it is just a technology.

Computing Technology

Data mining is an inter disciplinary approach having knowledge from different fields such as databases, statistics, high performance computing, machine learning, visualization and mathematics to automatically extract concepts, and to determine interrelations and patterns of interest from large databases.

Why Data Mining?

HUGE VOLUME THERE IS WAY TOO MUCH DATA & GROWING!

Data collected much faster than it can be processed or managed. NASA Earth Observation System (EOS), will alone, collect 15 Peta bytes by 2007 (15,000,000,000,000,000 bytes).

- Much of which won't be used - ever!
- Much of which won't be seen - ever!
- Why not?
- There's so much volume, usefulness of some of it will never be discovered

SOLUTION: Reduce the volume and/or raise the information content by structuring, querying, filtering, summarizing, aggregating, mining...

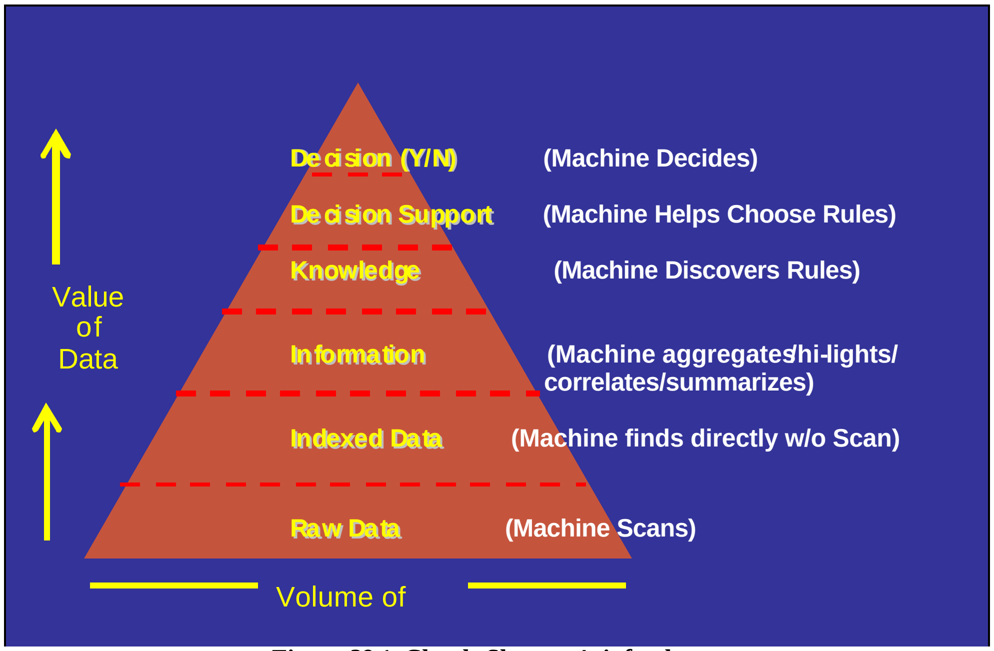
Data Mining is the exploratory data analysis with little or no human interaction using computationally feasible techniques, i.e., the attempt to find interesting structures/patterns unknown a priori

The ability of data mining techniques to deal with huge volumes of data is a distinguishing characteristic. The data volumes of organizations/enterprises/businesses are increasing with a greater pace e.g. NASA Earth Observing System (EOS) generates more than 100 gigabytes of image data per hour, stored in eight centers. Data collected much faster than it can be processed or managed. NASA EOS will alone, collect 15 Peta bytes by 2007 (15,000,000,000,000,000 bytes). Plunging in to such a deep and vast sea of data and performing analysis is not possible through conventional techniques. So majority of the data will remain untouched, unseen, unused thus limiting the discovery of useful patterns. This requires the availability of tools and techniques that can accommodate the huge volumes of data and its complexity. Obviously, data mining is the best fit. It works by reducing the data volume and/or raise the information content user interest by structuring, querying, filtering, summarizing, aggregating, mining etc. By raising information content means that the those patterns are built and brought to surface that are otherwise hidden and scattered in the data sea and of user interest.

Claude Shannon's info. theory

More volume means less information

Claude Shannon’s theory states that as the volume increases the information content decreases and vice versa.



Decision (Y/N)
(Machine Decides)
Decision Support
(Machine Helps Choose Rules)
Knowledge
(Machine Discovers Rules)
Value of Data
Information
(Machine aggregates/hi-lights/ correlates/summarizes)
Indexed Data
(Machine finds directly w/o Scan)
Raw Data
(Machine Scans)
Volume of

Figure-29.1: Claude Shannon's info. theory

The Figure 29.1 well illustrates the Shannon's Information Theory. At the base lies the raw data having maximum volume. Here the data value, that increases as we go up (volume decreases), is the minimum. Here exploring data for useful information needs conventional data scanning, thus one is lost in the deep blue sea, reaching no-where.

In the next level is the indexed data. Data indexing has greatly reduced the data volume as compared to the data in the lower level (raw data). Now we have found short cuts, to reach desired points in the voluminous data sea, rather than conventional scanning. The data has more value than the data at lower level.

In the next level is the aggregate/summarized data. Here the data is in a form that can readily be used as information. Thus much compact volume and greater data value than the previous lower levels.

Next is the level where the machine discovers and learns rules. The rules can easily be applied to extract only desired data, or knowledge base thus greatly reducing the data volume as compared to other lower levels.

The next is the level where machine supports decision making process by helping in selecting appropriate pre defined rules. Here data volume is minimal and similarly higher data value than the lower levels.

The final top most level is where the machine itself makes decisions based on predefined rules. Those most appropriate rules are selected by machine itself for making a decision.Here of course the data volume is minimum and the value of data is thus maximum.

Why Data Mining?: Supply & Demand

Amount of digital data recording and storage exploded during the past decade

BUT

number of scientists, engineers, and analysts available to analyze the data has not grown correspondingly.

Another reason of Data mining is that the data generation rate far exceeds the data capturing and analysis rate. In other words, the amount of digital data storage in different organizations world wide has greatly increased in the past few decades. However, the number of scientists, engineers and analysts for data analysis has not grown accordingly i.e. the supply of desired scientists and researchers don't meet their high demand in high numbers, so that the huge and continuously increasing data can be consumed or analyzed. Thus data mining tools provide a way, enabling limited scientists and researchers to analyze huge amounts of data.

Why Data Mining?: Bridging the gap

Requires solution of fundamentally new problems, grouped as follows:

1. developing algorithms and systems to mine large, massive and high dimensional data sets;
2. developing algorithms and systems to mine new types of data (images, music, videos);
3. developing algorithms, protocols, and other infrastructure to mine distributed data; and
4. improving the ease of use of data mining systems;
5. developing appropriate privacy and security techniques for data mining.

Data mining evolved as a mechanism to cater the limitations of OLTP systems to deal massive data sets with high dimensionality, new data types, multiple heterogeneous data resources etc. The conventional systems couldn't keep pace with the ever changing and increasing data sets. Data mining algorithms are built to deal high dimensionality data, new data types (images, video etc.) , complex associations among data items , distributed data sources and associated issues (security etc.)

Data Mining is HOT!

- **10 Hottest Jobs of year 2025**

Time Magazine, 22 May, 2000

- **10 emergi ng areas of technology**
  MIT's Magazine of Technology Review, Jan/Feb, 2001

The TIME Magazine May 2000 issue has given a list of the ten hottest jobs of year 2025. Data miners and knowledge engineers were at 5th and 6th position respectively. The proposed course/Curriculum will cover Data Mining. Hence Data mining is a hot field having wide market opportunities.

Similarly, MIT's Technology Review has identified 10 emerging areas of technology that will soon have a profound impact on the economy and how we live and work. Among the list of emerging technologies that will change the world, Data mining is at the 3rd place.

Thus in view of the above facts, *data miners* have a long career in national as well as international market as major companies both private and government are quickly adopting the technology and many have already adopted.

How Data Mining is different?

- **Knowledge Discovery**

--Overall process of discovering useful knowledge

- **Data Mining (**Knowledge-driven exploration**)**

-- Query formulation problem.
-- Visualize and understand of a large data set.
-- Data growth rate too high to be handled manually.

- **Data Warehouses** (Data-driven exploration):

-- Querying summaries of transactions, etc. ***Decision support***

- **Traditional Database (**Transactions**)**:

– Querying data in well-defined processes. ***Reliable storage***

Conventional data processing systems or online transaction processing systems (OLTP) lie at the bottom level. These systems have well defined queries and no any sort of knowledge discovery is performed. OLTP systems are meant to support day to day transactions and do not support decision making in any way. We can better understand with the analogy that when you travel from your home to university you may follow a same route very often. While on the way you will see same trees, same signals and same buildings every day provided you follow the same route. It is not possible that each and every day you see different buildings, trees than the previous day. Similar is the case for OLTP systems where you have well defined queries by running which you know what sort of results you can get. Nothing new or no discoveries are here.

Data Mining provides a global macroscopic view or aerial view of your data. You can easily see what you could not see at microscopic level. But before applying mining algorithms data must be brought in a form so that the knowledge exploration from huge, heterogeneous and multi source data can efficiently and effectively be performed. Thus DWH is the process of bringing input data in a form that can readily be used by data mining techniques to find hidden patterns. Both terns KDD and DM are sometimes used to refer to the same thing but KDD refers to the overall process from data extraction from legacy source systems, data preprocessing, DWH building, data mining and finally the output generation. So KDD is a mega process having sub processes like DWH and DM being its constituent parts.

How Data Mining is different...

Data Mining Vs. Statistics

- Formal statistical inference is assumption driven i.e. a hypothesis is formed and validated against the data.
- Data mining is discovery driven i.e. patterns and hypothesis are automatically extracted from data.
- Said another way, data mining is knowledge driven, while statistics is human driven.

Although both of the two are for data analysis and none is good or bad, some of the difference between statistics and Data mining are;

Statistic is assumption driven. A hypothesis is formed using the historical data and is then validated against current known data. If true the hypothesis becomes a model else the process is repeated with different parameters. DM, on the other hand, is discovery driven i.e. based on the data hypothesis is automatically extracted from the data. The purpose is to find patterns which are implicit and hidden in the data sea otherwise. Thus data mining is knowledge driven while statistics is human driven.

Data Mining Vs. Statistics

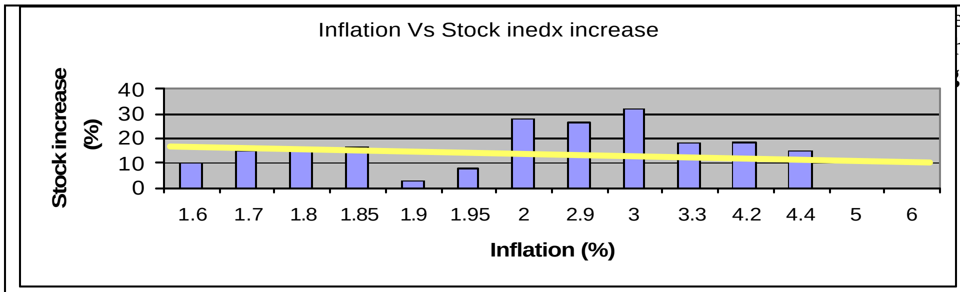
- Both resemble in exploratory data analysis, but statistics focuses on data sets far smaller than used by data mining researchers.
- Statistics is useful for verifying relationships among few parameters when the relationships are linear.
- Data mining builds much complex, predictive, nonlinear models which are used for predicting behavior impacted by many factors.

One difference is on the type of data. While statistician traditionally work with smaller and first hand data" that has been collected or produced to check specific hypotheses, data miners work with huge and second hand data" often assembled from different sources. The idea is to find interesting facts and potentially useful knowledge hidden in the data and often unrelated to the primary purpose why the data have been collected.

Statistics provides a language and framework for quantifying the uncertainty that results when one tries to infer general patterns from a particular sample of an overall population. The concern arose because if one searches long enough in any data set (even randomly generated data), one can find patterns that appear to be statistically significant but, in fact, are not.

Statistics is useful only for data sets with limited parameters (dimensions) and simple relationships (linear). Statistical methods fail when the data dimensionality is greater and the relationships among different parameters are complex. Data mining proves to be viable solution in such situations.

em
he
e



Q: What will be the stock increase when inflation is 6%?

A: Model non -linear relationship using a line *y* = *mx* + *c*. Hence answer is 13%

Figure-29.2: Knowledge extraction using statistics

What statistics can offer for knowledge discovery? Consider the histogram in Figure 29.2. It shows the relationship between %inflation and %stock increase. What if we want to know the value of %stock increase at 6% inflation? W can use linear regression as shown by yellow line in the figure. The line acts as a conducting wire, balanced between magnets (bars). Thus regression here is like balancing of a wire, so that the final position of the line is as shown in Figure 29.2. Thus we use curve fitting with linear regression and the line equation $y=mx+c$ to calculate the value at 6% inflation. Here m is the slope x is known 6% and c is the intercept.

Realistically only two variables here, however, in real life many variable may be more than 50. In such situations linear regression, the statistical methods fail.

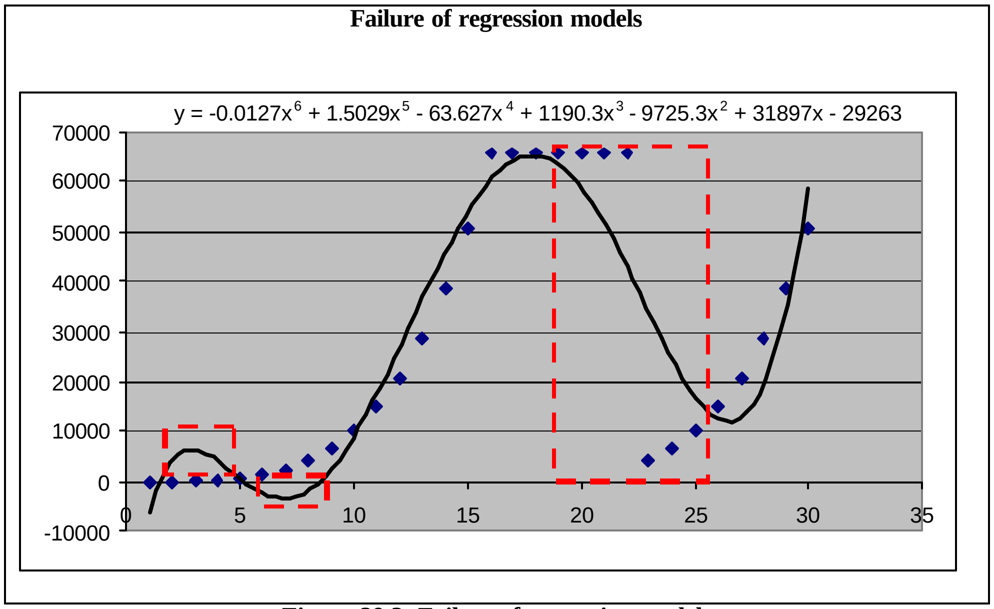
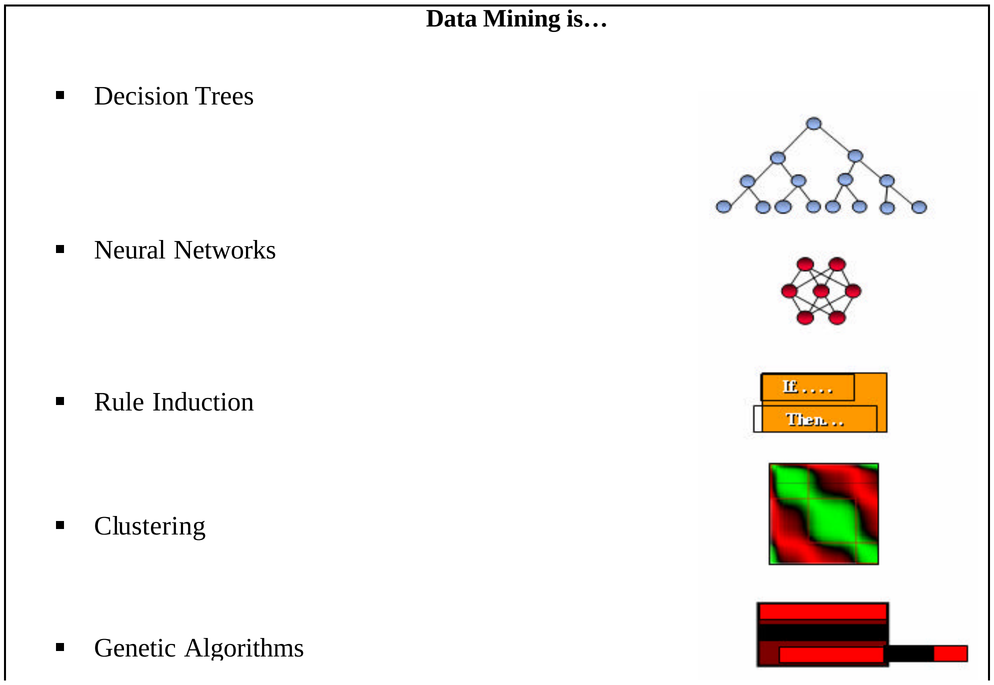


Figure-29.3: Failure of regression models

To better understand the limitation of regression, consider a real life example in Figure 29.3. Although l two variables here too, but the relationship is not that much simple. In real life the association between variables is not that much simple rather complex as shown by the equation in Figure-29.3. Here dotted line is the actual data. The non linearity of the association leads to the dotted line structure having peaks and falls. Here we can't use linear regression like we did in our previous example. Here we do a polynomial curve fitting like degree 6 curve fitting shown by regular line in the figure. The red dotted boxes show the difference between the actual and the curve fitting. We can seen the difference is initially is less than what we can see in the next red doted box in the middle. Thus the crux is that regression just failed even for two variables with complex associations. How can it be applied to real life data having 20, 25 or even 50 attributes?



Data Mining is…

- Decision Trees
- Neural Networks
- Rule Induction
- Clustering
- Genetic Algorithms

Now lets discuss something about what is included in DM and what is not. First we will discuss what DM is.

Decision Trees (DT): Decision trees consist of dividing a given data set into groups based on some criteria or rule. The final structure looks like an inverted tree, hence the technique called DT. Suppose a table having a number of records. The tree construction process will group most related records or tuples in the same group. Decision at each node is taken based on some rule, if this then this else this. Rules are not known a priori and are digged out of the training data set.

Clustering: It is one of the most important Dm techniques; we will discuss it in detail in coming lectures. As a brief for understanding it involes the grouping of data items without taking any human parametric input. We don't know the number of clusters and their properties a priori. Two main types are one way clustering and two way clustering. One way clustering is when only data records (rows) are used. Two way clustering is when all the rows and columns are being used for clustering purpose.

Genetic Algorithms: These are based on the principle survival of the fittest. In these techniques, a model is formed to solve problems having multiple options and many values. Briefly, these techniques are used to select the optimal solution out of a number of possible solutions. However, are not much robust as can not perform well in the presence of noise.

Lecture-30
What Can Data Mining Do

Our previous lecture was a brief introduction about the data mining. What we covered in lecture 29 was just the tip of iceberg. The lecture may have definitely created in you an excitement to explore more about DM like me. So this lecture is meant to give you more insight of DM, what it can do for us, what are its specific applications. I will give some real life examples to show the power of DM, what are the problems that can be solved by DM.

There are a number of data mining techniques and the selection of a particular technique is highly application dependent, although other factors affect the selection process too. So let's look at some of the DM application are as or techniques.

CLASSIFICATION

- Classification consists of examining the properties of a newly presented observation and assigning it to a predefined class.
  - Assigning customers to predefined customer segments (good vs. bad)
  - Assigning keywords to articles
  - Classifying credit applicants as low, medium, or high risk
  - Classifying instructor rating as excellent, very good, good, fair, or poor

Classification means that based on the properties of existing data, we have made or groups i.e. we have made classification. The concept can be well understood by a very simple example of student grouping. A student can be grouped either as good or bad depending on his previous record. Similarly an employee can be grouped as excellent, good, fair etc based on his tra ck record in the organization. So how students or employees were classified? Answer is using the historical data. Yes history is the best predictor of the future. When an organization conducts test and interviews from candidate employees, their performance is compared with those of the existing employees. The knowledge can be used to predict how good you can perform if employed. So we are doing classification, here absolute classification i.e. either good or bad or in other words we are doing binary classification. Either you are in this group or this. Each entity is assigned one of the groups or classes. An example where classification can prove to be beneficial is in customer segmentation. The businesses can classify their customers as either good or bad; the knowledge thus can be utilized for executing targeted marketing plans. Another example is of a news site, where there are number of visitors and also many content developers. Now where to place a specific news item on the web site? What should be the hierarchical position of the news item, what should be the news chapter, category? Either it should be in the sports or weather section and so on. What is the problem in doing all this? The problem is that it's not a matter of placing a single news item. The site as already mentioned contains a number of content developers and also many categories. If sorting is performed humanly, then it is time consuming. That is why classification techniques can scan and process the document to decide its category or class. How and what sort of processing will be discussed in the next lecture. It is not possible and there are flaws in assigning category to any news document just based on the keyword. Frequent occurrence of the word keyword cricket in a document doesn't necessary means that the document be placed in the sports category. The document may be actually political in nature.

ESTIMATION

As opposed to discrete outcome of classification i.e. YES or NO, deals with continuous valued outcomes

Example:

Building a model and assigning a value from 0 to 1 to each member of the set.

Then classifying the members into categories based on a threshold value.

As the threshold changes the class changes.

Next category of problems that can be solved with DM is using estimation. In classification we did binary assignment i.e. data items are assigned to either of the two categories or classes, this or that. The assignment value was integer in nature, and to be absolute was not essential. However in case of estimation a model/mechanism is formed then data analysis is performed in that model which was actually formed from data itself. The difference is that the model is formed from the relationships in the data and then data categorized in that model. Unlike classification, categorization here is not absolute but there is a real number that is between 0 and 1. This number tells the probability of a record/tuple/item etc. to belong to a particular group or category or class. So this is a more flexible approach than classification. Now the question arises how a real number between 0 and 1 reveals the probability of belonging to a class? Why not an item falls in two groups or more at the same time? The answer is that categorization is performed by setting the threshold values. It is predefined that if the value crosses this then in this class else in another class and so on. Note that if thresholds are reset which is possible (as nothing is constant except change so changes can be made in the threshold) then the category or class boundaries change resulting in the movement of records and tuples in other groups accordingly.

PREDICTION

Same as classification or estimation except records are classified according to some predicted future behavior or estimated value.

Using classification or estimation on a training example with known predicted values and historical data a model is built.

Then explain the known values, and use the model to predict future.

Example:

Predicting how much customers will spend during next 6 months.

Prediction here is not like a palmists approach that if this line then this. Prediction means that what's the probability of an item/event/customer to go in a specific class. This means that prediction tells that in which class this specific item would lie in future or to which class this specific event can be assigned in any time in future, say after six years. How prediction actually works? First of all a model is built using exiting data. The existing data set is divided into two subsets, one is called the training set and the other is called test set. The training set is used to form model and the associated rules. Once model built and rules defined, the test set is used for grouping. It must be noted the test set groupings are already known but they are put in the model to test its accuracy. Accuracy, we will discuss in detail in following slides but is dependent on many factors like the model, training data and test data selection and sizes and many more things.

So, the accuracy gives the confidence level, that the rules are accurate to that much level. Prediction can be well understood by considering a simple example. Suppose a business wants to know about their customers their propensity to buy/spend/purchase. In other words, how much the customer will spend in next 6 months? Similarly a mobile phone company can install a new tower based on the knowledge spending habits of its customers in the surroundings. It is not the case that companies install facilities or invest money because of their gut feelings. If you think like this you are absolutely wrong. Why companies should bother about their customers? Because if they know their customers, their interests, their like and dislikes, their buying patterns then it is possible to run targeted marketing campaigns and thus increasing profit.

MARKET BASKET ANALYSIS

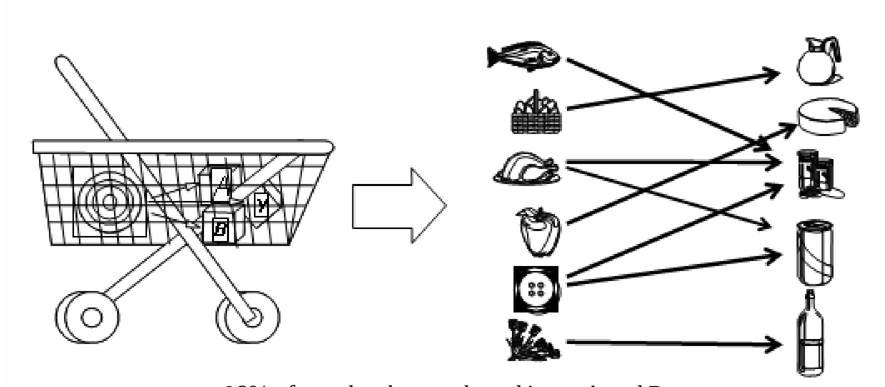
Determining which things go together, e.g. items in a shopping cart at a super market.

Used to identify cross-selling opportunities

Design attractive packages or groupings of products and services or increasing price of some items etc.

Next problem can be solved or being solved by DM is the market basket analysis. Market basket analysis is a concept, like in big stores you may have seen baskets for roaming around and putting selected items in it. The concept here is to know basically that which things are sold together. Why the knowledge is needed for decision making? This can be beneficial because if we know that these are the things which are sold together, or if we know that some type of customers mostly buy item X too when they buy item Y and so on, then we can run corresponding sale promotional schemes. This can be useful for inventory management i.e. you may place things that are bought together in close proximity, or you can place those things close in your store so that it's easy to bring things together when needed. Another benefit is that u can bundle or package item together so as to boost the sales of underselling items. Customer satisfaction is critical for businesses, so another benefit of market basket analysis is the customer facilitation. Thus, by placing together the associated items you facilitate the customer making easy access to the desired items. Otherwise he may rum here and there for searching the item.

MARKET BASKET ANALYSIS



98% of people who purchased items A and B also purchased item C

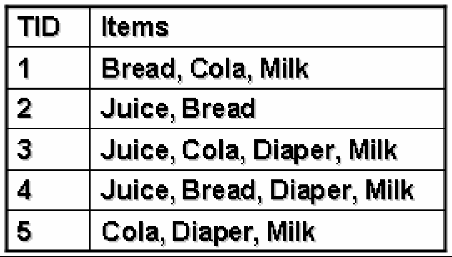
Figure-30.1: Market basket analysis

Lets consider an example for better understanding of the market basket analysis. Figure 30.1 shows a basket having items A, B and Y. The right side portion of the Figure shows different items arranged in two columns and arrows show the associations i.e. if item in the left column is bought then the respective item in the right column is also bought. How we came to know this? This knowledge was hidden deep in the data, so querying was not possible because a store may contain thousands of items and effort to find item associations trivially is impossible, an NP complete problem.

Discovering Association Rules

- Given a set of records, each containing set of items
    - Produce dependency rules that predict occurrences of an item based on others

- Applications:
    - Marketing, sales promotion and shelf management
    - Inventory management



TID	Items
1	Bread, Cola, Milk
2	Juice, Bread
3	Juice, Cola, Diaper, Milk
4	Juice, Bread, Diaper, Milk
5	Cola, Diaper, Milk

Rules:
{Milk} $\rightarrow$ {Cola}
{Diaper, Milk} $\rightarrow$ {Juice}

Table -30.1: Discovering association rules

Discovering Association Rules is another name given to market basket analysis. Here rules are formed from the dependencies among data items which can be used to predict the occurrence of an item based on others e.g. suppose hardware shop where whenever a customer buys color tins it is more likely that he /she will buy painting brushes too. So based on the occurrence or event of paint purchase, we can predict the occurrence of item paint brush. What is the benefit of knowing all this? We have already discussed this. Now look at the Table 30.1, here two columns TIC (transaction ID) and other is the list of items. This is not the view of a real database, as a single column can not contain multiple entries like items column here. This is an example table just to show rule formation process. Looking at the table we come to know that whenever milk is purchased, cola is also purchased. Similarly, whenever diaper and milk are purchased juice is also purchased. So, the two association rules are obtained from the sample data in Table 30.1. Now a question arises which of the two rules strongly implies? This can not be answered depending on a lot of factors. However, we can tell what has been discovered here? What is the unknown unknown? The discovery is that the sale of juice with diapers and milk is non trivial. This can never be guessed because no obvious association is found among the items.

CLUSTERING

Task of segmenting a heterogeneous population into a number of more homogenous sub-groups or clusters.

Unlike classification, it does NOT depend on predefined classes.

It is up to you to determine what meaning, if any, to attached to resulting clusters.

It could be the first step to the market segmentation effort.

What else data mining can do? We can do clustering with DM. Clustering is the technique of reshuffling, relocating exiting segments in given data which is mostly heterogeneous so that the new segments have more homogeneous data items. This can be very easily understood by a simple example. Suppose some items have been segmented on the basis of color in the given data. Suppose the items are fruits, then the green segment may contain all green fruits like apple, grapes etc. thus a heterogeneous mixture of items. Clustering segregates such items and brings all apples in one segment or cluster although it may contain apples of different colors red, green, yellow etc. thus a more homogeneous cluster than the previous cluster.

Clustering is a difficult task, why? In case of classification we already know the number of classes, either good or bad or yes or no or any number of classes. We also have the knowledge of classes properties so its easy to segment data into known classes. However, in case of clustering we don't know the number of clusters a priori. Once clusters are found in the data business intelligence, domain knowledge is needed to analyze the found clusters. Clustering can be the first step towards market segmentation i.e. we can use countermining to know the possible clusters in the data. Once clusters found and analyzed classification can be applied thus gaining more accuracy than any standalone technique. Thus clustering is at higher level than classification not only because of its complexity but also because it leads to classification.

Examples of Clustering Applications

- Marketing: Discovering distinct groups in customer databases, such as customers who make lot of long-distance calls and don't have a job. Who are they? Students. Marketers use this knowledge to develop targeted marketing programs.
- Insurance: Identifying groups of crop insurance policy holders with a high average claim rate. Farmers crash crops, when it is "profitable".
- Land use: Identification of areas of similar land use in a GIS database.
- Seismic studies: Identifying probable areas for oil/gas exploration based on seismic data.

We discussed that what clustering is and how it works. Now to know the real spirit of it, lets look at some of the real world examples to show the blessings of clustering;

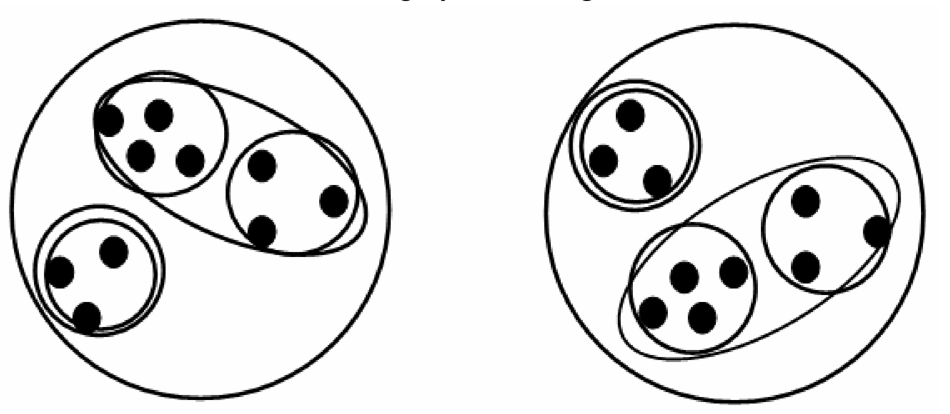
1. **Knowing or discovering about your market segment:** Suppose a telecom company whose data when clustered revealed that there is a group or cluster of people or customers whose

long distance calls are greater in number. Is this a discovery that such a group exists? Nope not really. The real discovery is analyzing the cluster, the real fun part. Why these people are in a cluster? Is important to know. Analysis of the cluster reveals that all the people in the group are unemployed! How come it is possible that unemployed people are making expensive far distance calls? The excitement lead to further analysis which ultimately revealed that the people in the cluster were mostly students, students like you living away from home in universities , colleges and hostels. They are making calls back home. So this is a real example of clustering. Now the same question what is the benefit of knowing al this? The answer is customer is like an asset for any organization. To know the customer is crucial for any organization/company so as to satisfy the customer which is a key of any company's success in terms of profit. The company can rum targeted sale promotion and marketing effort to target customers i.e. students.

2. **Insurance:** Now lets have look at how clustering plays a role in insurance sector. Insurance companies are interested in knowing the people having higher insurance claim. You may astonish that clustering has successfully been used in a developed country to detect farmer insurance abuses. Some of the malicious farmers used to crash their crops intentionally to gain insurance money which presumably was higher than the amount of profit and effort from their crops. The farmer was happy but the loss was to be bear by the insurance company. The company successfully used clustering techniques to identify such farmers, and thus saving a lot of money.

Clustering thus has a wider scope in real life applications. Other areas where clustering is being used are for city planning, GIS (Land use management), seismic data for mining (real mining) and the list goes on.

Ambiguity in Clustering

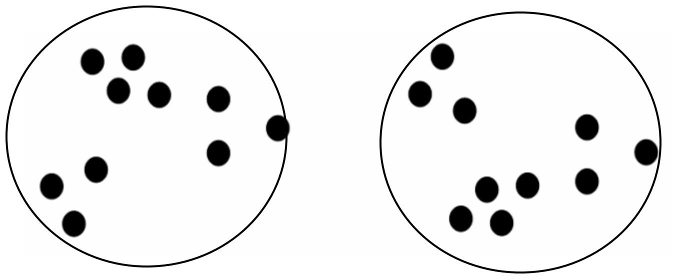


How many clusters?

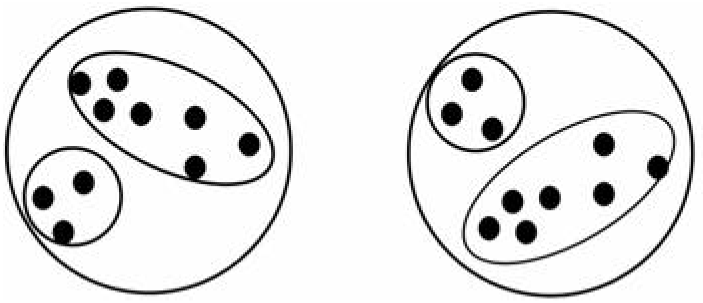
- **Two clusters**
- **Four clusters**
- **Six clusters**

Figure-30.2: Ambiguity in Clustering

As we mentioned the spirit of clustering lies in its analysis. A common ambiguity in clustering is regarding the number of clusters, since the cluster are not known in advance. To understand the problem, consider the example in Figure 30.2. The black dots represent individual data records or tuples and they are placed as a result of a clustering algorithm. Now can u tell how many clusters are there?



Yes two clusters, but look at your screens again and tell how many clusters now?



Yes four clusters now, you are absolutely right. Now look again and tell how many clusters? Yes 6 clusters as shown in the Figure 30.2. What all this shows? This shows that deciding upon the number of clusters is a complex task depending on factors like level of detail, application domain etc. By level of detail I mean that either the black point represents a single record or an aggregate. The thing which is important is to know how many clusters solve our problem. Understanding this solves the problem.

DESCRIPTION

Describe what is going on in a complicated database so as to increase our understanding.

A good description of a behavior will suggest an explanation as well.

Another application of DM is description. To know what is happening in our databases is beneficial. How? The OLAP cubes provide ample amount of information, which is otherwise distributed in the haystack. We can move the cube in different angles to get to the information of interest. However, we might miss the angle which might have given use some useful information. Description is used to describe such things.

Comparing Methods (1)

- *Predictive accuracy*: this refers to the ability of the model to correctly predict the class label of new or previously unseen data
- *Speed*: this refers to the computation costs involved in generating and using the method.
- *Robustness*: this is the ability of the method to make correct predictions/groupings given noisy data or data with missing values

We discussed different data mining techniques. Now the question, which technique is good and which bad? Or say like which is the best technique for a given problem. Thus we need to specify evaluation criteria like data metrics as we did in the data quality lecture. The metrics we use for comparison of DM techniques are;

Accuracy: Accuracy is the measure of correctness of your model e.g. in classification we have two data sets, training and test sets. A classification model is built based on the data properties and relationships in training data. Once built the model is tested for accuracy in terms of % correct results as the classification of the test data is already known. So we specify the correctness or confidence level of the technique in terms % accuracy.

Speed: In previous lectures we discussed the term "Need for Speed". Yes speed is a crucial aspect of Dm techniques. Speed refers to the time complexity. If a technique has O (n) and another has O (n log n) time complexities then which is better? Yes O (n) is better. This is the computational time but user or business decision maker is interested in the absolute clock time. He has nothing to do with complexities. What he is interested in is, knowing how fast he gets the answers. So just comparing on the basis of complexities is not sufficient. We must look at the overall process and interdependencies among tasks which ultimately result in the answer or information generation.

Robustness: It is the ability of the technique to work accurately even in conditions of noisy or dirty data. Missing data is a reality and presence of noise also true. So a technique is better if it can run smoothly even in stress conditions i.e. with noisy and missing data.

Scalability: As we mentioned in our initial lectures that the main motivation for data warehousing is to deal huge amounts of data. So scaling is very important, which is the ability of the method to work efficiently even when the data size is huge.

Interpretability: It refers to the level of understanding and insight that is provided by the method. As we discussed in clustering one of the complex and difficult tasks is the cluster analysis. The techniques can be compared on the basis of their interpretational ability e.g. there might be some methods which give additional functionalities to provide meaning to the discovered information like color coding, plots and curve fittings etc.

Where does Data Mining fits in?

Data Mining is one step of Knowledge Discovery in Databases (KDD)

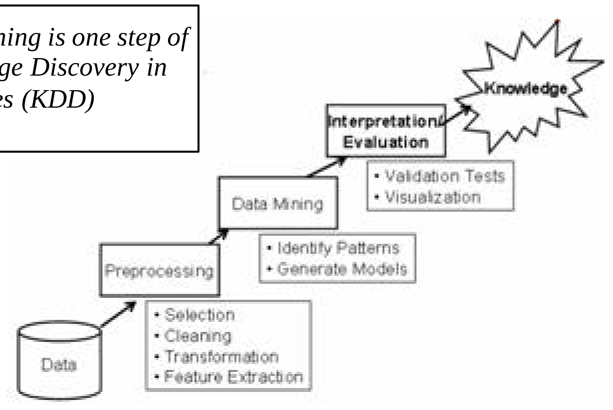


Figure-30.3: Where does Data Mining fits in?

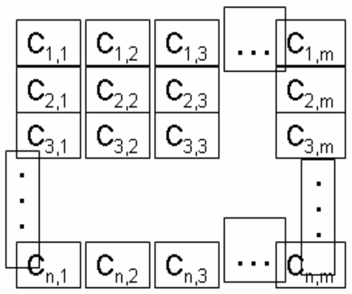
Now lets look at the overall knowledge discovery KD process. Figure 30.3 shows different KDD steps. Data is crucial and the most important component of KDD, knowledge discovery is possible only if data is available. The data is not used in its crude form for knowledge discovery. Before applying analysis techniques like data mining, data is preprocessed using activities as discussed in ETL. You can see an additional process at the preprocessing step i.e. feature extraction. This is to extract those data items which are needed or which or of interest form the huge data set. Suppose we have an O (n^2) method for data processing. Scalability can be issue for large data sets because of quadratic nature. The problem can be solved if we perform aggregation, reducing number of records and keeping the data properties. So now the method can work with less data size. Next comes the data mining phase, we have thoroughly discussed the techniques for discovering patterns (clustering) and generating models (classification). The next step is the analysis of discovered patterns using domain knowledge. This is a complex task and requires an ample amount of business or domain knowledge. The interpreted knowledge finally comes out as the information that was unknown before hidden in the data sea which has now become information having some value to the user.

Lecture# 31
Supervised Vs. Unsupervised Learning

In the previous lecture we discussed briefly DM concepts. Now we look with some greater details, two main DM methods supervised and unsupervised learning. Supervised learning is when you are performing DM the supporting information that helps in the DM process is also available. What could be that information? You may know your data that how many groups or classes your data set contains. What are the properties of these classes or clusters? When we will talk about unsupervised learning you will not have such known or a priori knowledge. In other words you can not give such factors as input to the DM technique which can facilitate your DM process. So wherever the user gives some input that is supervised else that is unsupervised learning.

Data Structures in Data Mining

- Data matrix
  - Table or database
  - *n* records and *m* attributes,
  - *n* >> *m*


$C_{1,1}$ $C_{1,2}$ $C_{1,3}$ ... $C_{1,m}$
$C_{2,1}$ $C_{2,2}$ $C_{2,3}$ $C_{2,m}$
$C_{3,1}$ $C_{3,2}$ $C_{3,3}$ $C_{3,m}$
$C_{n,1}$ $C_{n,2}$ $C_{n,3}$ ... $C_{n,m}$

- Similarity matrix
  - Symmetric square matrix
  - *n* x *n* or *m* x *m*

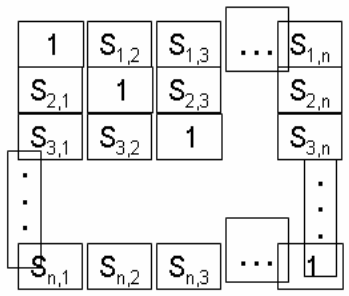
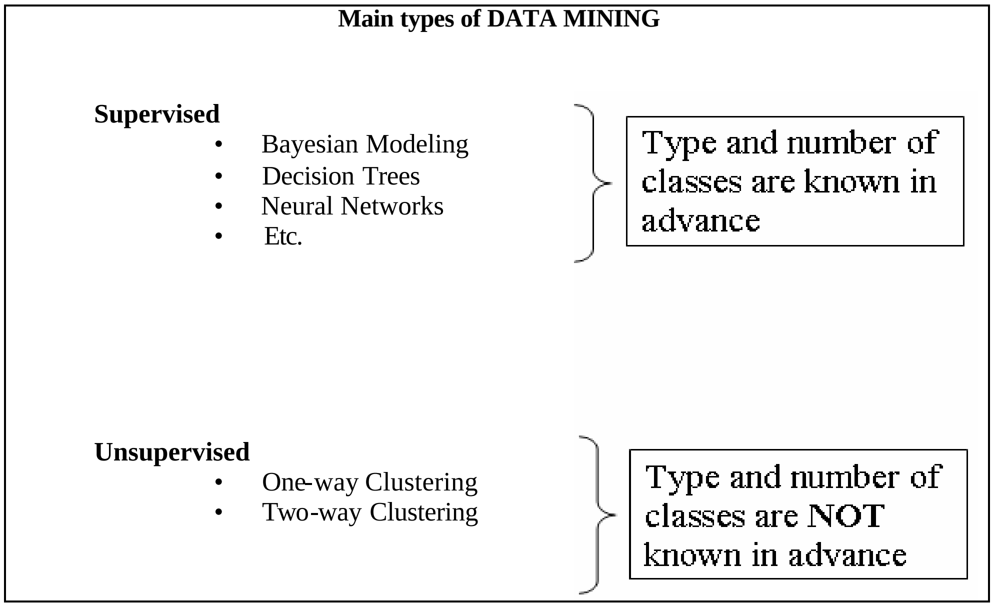

1 $S_{1,2}$ $S_{1,3}$ ... $S_{1,n}$
$S_{2,1}$ 1 $S_{2,3}$ $S_{2,n}$
$S_{3,1}$ $S_{3,2}$ 1 $S_{3,n}$
$S_{n,1}$ $S_{n,2}$ $S_{n,3}$ ... 1

Figure 31.1: Data Structures in data mining

First of all we will talk about data structures in DM. What does DS mean here? You can consider it as pure data structure but we specifically mean how you store your data. Figure 31.1 shows two metrics data matrix and the similarity matrix. Data matrix means the table or database used as the input to the DM algorithm. What will be the dimensions or size of that table normally? The size of records (rows) is much greater than the number of columns. The attributes may be 10, 15 or 25 but the number of rows far exceeds the number of columns e.g. a customer table may have 20-25 attributes but the total records may be in millions. As I said previously that the mobile users in Pakistan are about 10 million. If a company even has 1/3 of the customers then 3.3 *lakh* customer records in the customer table. Thus greater number of rows than columns and there will be indices *i* and *j* in the table and you can pick the particular contents of a cell by considering the intersection of the two indices.

The second matrix in the Figure 31.1 is called the similarity matrix. Lets talk about its brief background. Similarity or dissimilarity matrix is the measure the similarity or dissimilarity obtained by pair wise comparison of rows. First of all you measure the similarity of the row1 in data matrix with itself that will be 1. So 1 is placed at index 1, 1 of the similarity matrix. Then you compare row 1 with row 2 and the measure or similarity value goes at index 1, 2 of the similarity matrix and son. In this way the similarity matrix is filled. It should be noted that the similarity between row1 and row2 will be same as between row 2 and 1. Obviously, the similarity matrix will then be a square matrix, symmetric and all values along the diagonal will be same (here 1). So if your data matrix has *n* rows and *m* columns then your similarity matrix will have *n* rows and *n* columns. What will be the time complexity of computing similarity/dissimilarity matrix? It will be O (n^2) (m), where m accounts for the vector or header size of the data. Now how to measure or quantify the similarity or dissimilarity? Different techniques available like Pearson correlation and Euclidean distance etc. but in this lecture we have used Pearson correlation which you might have studied in your statistics course.



Main types of DATA MINING

Supervised

- Bayesian Modeling
- Decision Trees
- Neural Networks
- Etc.

Type and number of classes are known in advance

Unsupervised

- One-way Clustering
- Two-way Clustering

Type and number of classes are NOT known in advance

Now we will discuss the two main types Dm techniques as briefed in the beginning. First we will discuss supervised learning which includes Bayesian classification, decision trees, neural networks etc. Lets discuss Bayesian classification or modeling very briefly. Suppose you have a data set and when you process that data set, say when you do profiling of your data you come to know about the probability of occurrence of different items in that data set. On the basis of that probability, you form a hypothesis. Next you find the probability of occurrence of an item in the available data set on that hypothesis. Similarly, how this can be used in decision trees? To understand suppose there is insurance company and is interested in knowing about the risk factors. If a person is of age between 20 and 25, he is unmarried and his job is unstable then there is a risk in offering insurance or credit card to such a person. This is because if married one may drive carefully even thinking of his children than otherwise. Thus when the tree was formed the classes, low risk and high risk were already known. The attributes and the properties of the classes were also known. This is called supervised learning.

Now unsupervised learning where you don't know the number of clusters and obviously no idea about their attributes too. In other words you are not guiding in any way the DM process for performing the DM, no guidance and no input. Interestingly, some people say their techniques are unsupervised but still give some input although indirectly. So a pure unsupervised algorithm is the one which don't have any human involvement or interference in any way. However, if some information regarding the data is needed, the algorithm itself can automatically analyze and get the data attributes. There are two main types of unsupervised clustering.

1. **One-way Clustering**-means that when you clustered a data matrix, you used all the attributes. In this technique a similarity matrix is constructed, and then clustering is performed on rows. A cluster also exists in the data matrix for each corresponding cluster in the similarity matrix.

2. **Two-way Clustering/Biclustering**-here rows and columns are simultaneously clustered. No any sort of similarity or dissimilarity matrix is constructed. Biclustering gives a local view of your data set while one-way clustering gives a global view. It is possible that you first take global view of your data by performing one-way clustering and if any cluster of interest is found then you perform two-way clustering to get more details. Thus both the methods complement each other.

Clustering: Min-Max Distance

Finding groups of objects such that the objects in a group are similar (or related) to one another and dissimilar from (or unrelated to) the objects in other groups e.g. using K-Means.

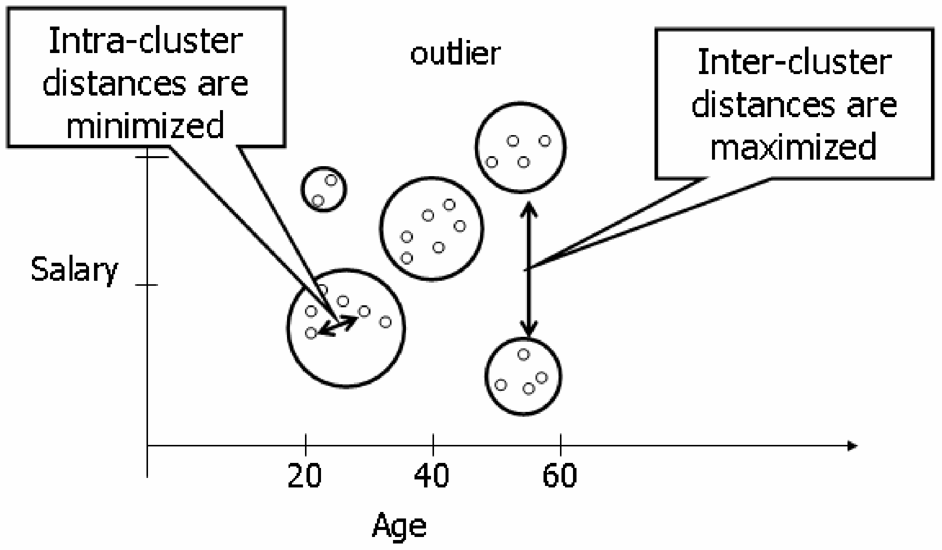

Intra-cluster distances are minimized
outlier
Inter-cluster distances are maximized
Salary
20
40
60
Age

Figure-31.2: Min-Max Distance

Now we talk about "distance" which here means that there must be some measure of telling how much similar or dissimilar a record is form another e.g. height, GPA salary are examples of a single attribute. So there should be a mechanism of measuring the similarity or dissimilarity. We have already discussed clustering ambiguities, so when clusters are formed in unsupervised clustering, the points having smaller intracluster distances should fall in a single cluster and inter clustering distance between any two clusters should be greater. For example, consider the age and salary attributes of employees as shown in Figure 31.2. Every point here corresponds to a single employee i.e. a single record. Look at the cluster around age 60, these might be retired people getting pensions only. There is another cluster at the age of 68, these are the people aged enough but still may stick to their organizations. The thing to understand here is that as the age increases the salary increases but when people get retirements their salaries fall. The first cluster in the diagram shows young people with low salaries. There is another cluster too that shows old people with low salaries. There is also a cluster showing young and high salary, called outliers. These are the people who might be working in their "Abba Jee's "company. So u must have understood outliers, inter-cluster and intra-cluster distances.

How Clustering works?

- Works on some notion of natural association in records (data) i.e. some records are more similar to each other than the others.
- This concept of association must be translated into some sort of numeric measure of the degree of similarity.
- It works by creating a similarity matrix by calculating pair-wise distance measure for *n x m* data matrix.
- Similarity matrix if represented as a graph can be clustered by:
  - Partitioning the graph
  - Finding cliques in the graph

In clustering there has to be the notion of association between records i.e. which records are associated with other records in the data set. You have to find that association. There has to be a distance matrix and you have to map that on the association. You may be able to quantify the association among closely related records. Remember that all the attributes may not be numeric, attributes may be non numeric as well e.g. someone's appointment, job title, likes and dislikes are all non numeric attributes. Thus taking these numeric and non numeric attributes you have to make an association measure which will associate records. When such a measure is formed, it will reflect the similarity between records. On the basis of that similarity a similarity matrix will be built (one way clustering) that will be square, symmetric having same diagonal value 1. Since we are using Pearson's coefficient the values in the matrix will be between -1 and +1. For opposing items when one is rising and the other is falling, the correlation will be -1. For those items that fall and rise simultaneously, there will be a correlation of +1 and if no correlation then 0. For the sake of simplicity, it is represented as a binary matrix i.e. if correlation greater than 0.5 it will be represented with 1. If correlation less than 0.5, it will be represented with 0. Thus in this way the similarity matrix which was real is transformed into the binary matrix. Question arises where is the clustering? Now assume that your binary matrix represents a graph. As you might have studied that a graph can be represented as a matrix. So we utilize some graph properties for performing clustering. We can use two techniques. One is the graph partitioning in which the graph is portioned in such a way that the connectivity among vertices in the partition is higher than across the partitions. Thus we performed clustering through graph partitioning. Another technique called clique detection can also be used but we will not go into the details.

One-way clustering

Clustering all the rows or all the columns simultaneously.

Gives a global view.

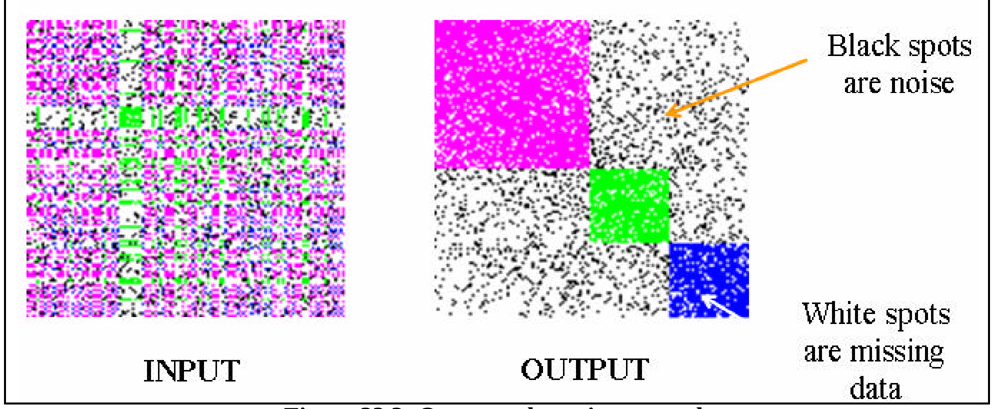


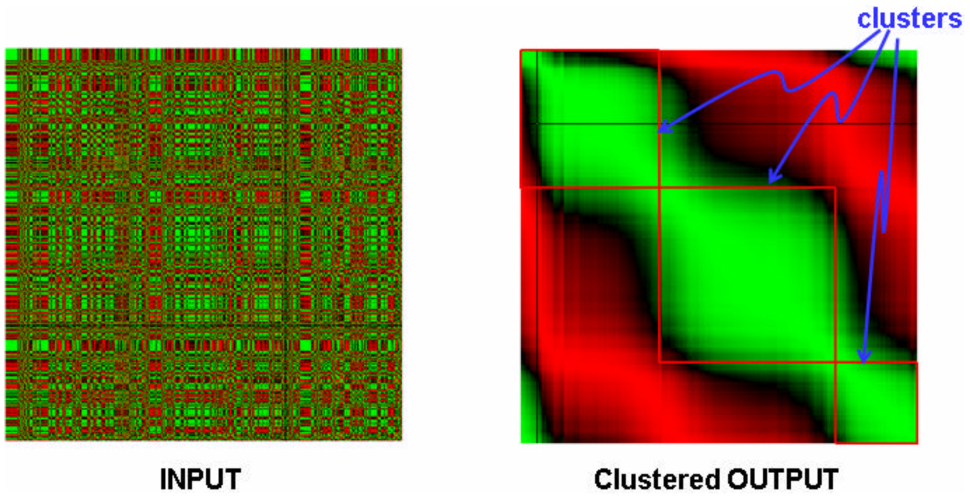
Figure 32.3: One-way clustering example

Now I will give you a real example of one way clustering. A data matrix is taken and intentionally put 3 clusters in that. Then similarity matrix is formed from data matrix and the clusters are also color coded for better understanding. The similarity matrix is then randomly permuted and some also is also inserted. The final shape of the matrix is as shown in Figure 32.3 and will be used as input to the clustering technique. You can see some black and white points. White points represent missing values which can either be in data or similarity matrix. Black points represent noise i.e. those data values which should not be present in the data set. Now I apply my own indigenous technique for performing clustering this input similarity matrix. The result is shown as output in Figure 32.3. Here you can see very well defined clusters and all these clusters are clearly visible and distinguishable because of color coding. Remember that in previous lecture it was stated that good DM techniques should be robust. So our technique applied here is highly robust because it worked fairly well even in the presence of noise. Genetic algorithms that we discussed previously are not robust because they do not work well in the presences of noise.

Data Mining Agriculture data

Created a similarity matrix using farm area, cotton variety and pesticide used.

Color coded the matrix, with green assigned to +1 and red to -1.



Large farmers using fewer varieties of cotton and fewer pesticides.

Small farmers using left-overs.

Figure-32.4: Data Mining Agriculture data

In the previous slide the example was based on simulated data where the clusters were intentionally inserted into the data. Now we will consider a real world example where agricultural data has been use. The data, pest scouting data, has been taken from department of pest warning Punjab for the year 2000 and 2001. Some of the attributes like farm area, cotton variety cultivated and pesticide used are selected. Using these and some other attributes a similarity matrix has been formed based on Pearson's Correlation. The matrix thus has values between +1 and -1 for better understanding color coding is used and most positive i.e. +1 is represented with green and -1 is represented with red color. Furthermore, as these values proceed towards zero from either side, the colors are darkened accordingly.

Input matrix in Figure 32.4 shows the color coded similarity matrix. We can say that data visualization has been performed here since we can see the relationship among records which are around 600 to 700 in number. Otherwise it is not possible to visualize such a big table in a single sight. Now when the clustering technique is applied in this input matrix the result is shown in Figure 32.4 as clustered output. Here we see that the green points have grouped together i.e. those points having high correlation or records having more association have grouped together. This is what we needed and points with low association i.e. red color points have separated. Here one thing is important that the input and the output matrices are identical except that the order of rows and columns has been changed. The cluster in the output matrix is also present in the input matrix but is not visible only because of reordering. Interesting enough if you search the world's greatest and number one search engine **Google** for key words “data mining” and “agriculture”, it results in *4 lakh* hits and 4th hit being the work done by me.

So what is the significance of doing all what mentioned above? The results showed that which type of farmers small or big use what sort of varieties, which pesticide and in what quantity etc. The information is useful from marketing point of view and can be used by pesticide manufacturing companies, seed breeders and so on for making effective decisions about their target market.

Classification

Examining the features of a newly presented object and assigning it to one of a predefined set of classes.

Example:
Assigning keywords to articles. Classification of credit applicants. Spotting fraudulent insurance claims.

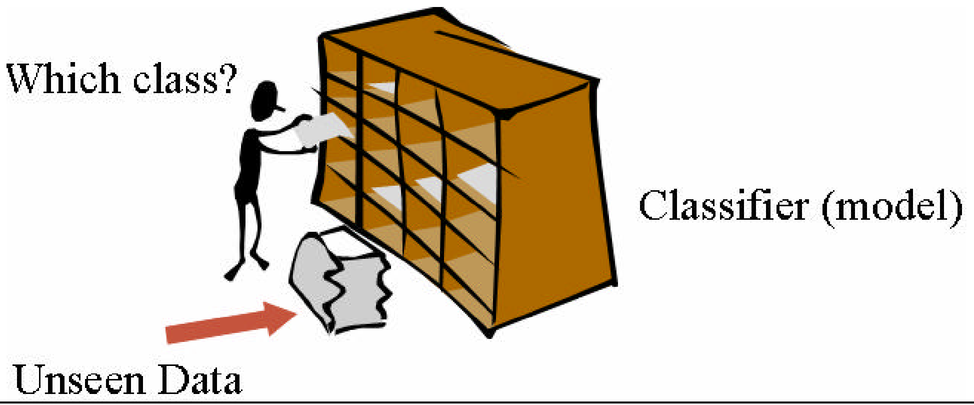


Figure-32.5: Classification

In one of the previous slide took and overview of classification. Now we will discuss it in detail but before this have a look at the analogy in Figure 32.5. Here you can see a person is standing by a rack that is boxed. It may also be called as a *pigeon hole box.* Each box in the rack represents a class while the whole rack is a classifier model. Alongside the rack, there also lies a documents filled bag which is the unseen data. It means that it is unknown that to which box or class these documents belong. So using the classification technique and the built in model each document is assigned to its respective box or class in the rack. Thus for performing classification, you must have a classification model in place and also the classes must be known a priori. You must know the number of classes and there attributes as well i.e. by looking at the data properties of any picked document, the classification process will know the box/class of the document that it belongs to. Thus in this way classification is performed. The classification can be used for customer segmentation, to detect fraudulent transactions and issues related to money laundering and the list goes on.

How Classification work?

- It works by creating a model based on known facts and historical data by dividing into training and test set.
- The model is built using the test set where the class is known and then applying it to another situation where the class is unknown.

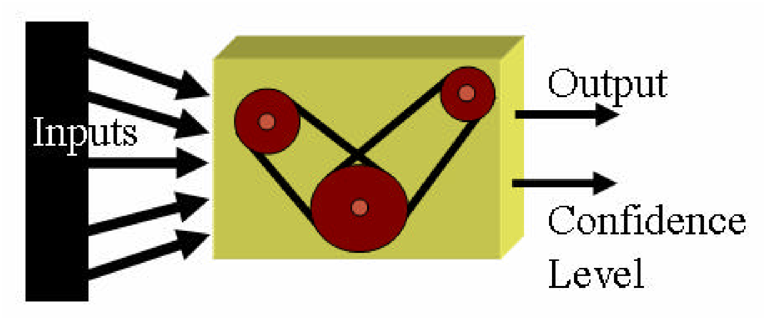
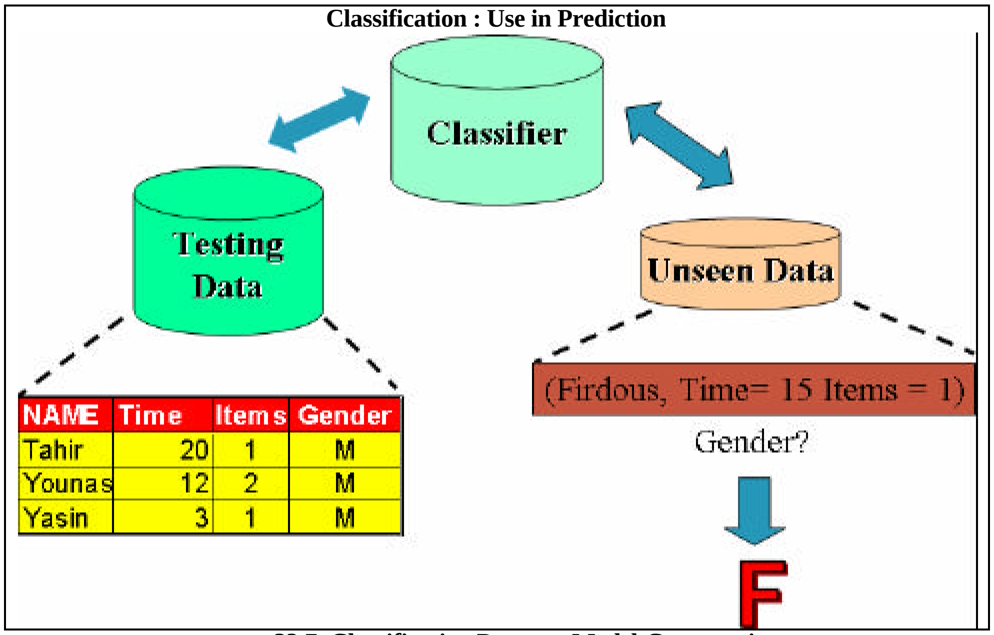


Figure-32.6: How Classification works?

With understanding of classification by analogy in previous slide, lets discuss formally how the classification process actually works. First of the available data set is divided into two parts, one is called test set and the other is called the training set. We pick the training set and a model is constructed based on known facts, historical data and class properties as we already know the number of classes. After building the classification model, every record of the test set is posed to the classification model which decides the class of the input record. Thus class label is given as the output of the model as shown in Figure 32.6. It should be noted that you know the class for each record in test set and this fact is used to measure the accuracy or confidence level of the classification model. You can find accuracy by

Accuracy or confidence level = matches/ total number of matches

In simple words, accuracy is obtained by divid ing number of correct assignments by total number of assignemnets by the classification model.


Classification : Use in Prediction
Classifier
Testing Data
Unseen Data
(Firdous, Time= 15 Items = 1)
Gender?
F

NAME	Time	Items	Gender
Tahir	20	1	M
Younas	12	2	M
Yasin	3	1	M

gure-32.7: Classification Process- Model Construction

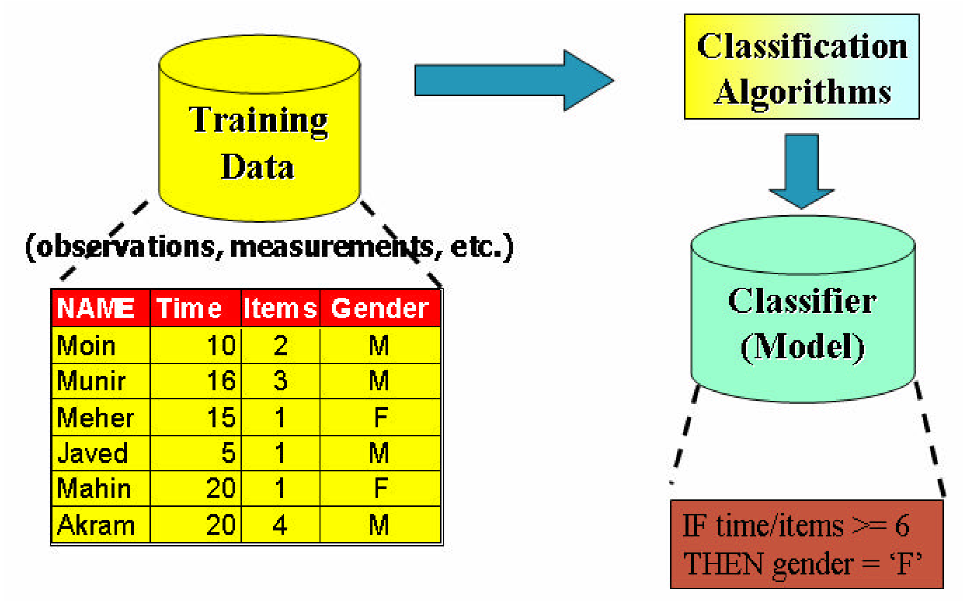
Here we will discuss how classification can be used for prediction. Before going into the details we must try to understand what if we predict? Does data mining really beneficial? In one of our previous lectures, we talked about assigning priorities to data sets in the context of data quality. We also talked about group priorities and group precedence. The concept was that if your data comes from two different sources and in one of those sources customers have told their gender and if data passes through right processes then you the customer gender (high data quality). However, there might be some customers which might not have specified or mentioned their gender. So the quality of such a data is questionable.

A possible solution to the problem of missing gender could be to assign gender on the basis of names. However, there might be some confusing names on the basis of which gender can not be assigned like Firdous, Riffat, and Shaheen etc. In such a situation we can use data mining for data cleansing. So lets have a look at how we form a classification model using a customer data set and then using that model for the identification of the unspecified gender which is mostly correct. So, two aspects data cleansing and classification are being covered here simultaneously.

First of all we will look in to the details of model construction process. Figure 32.8 shows a view of the training data set. Here you can see 6 records and 4 attributes. Time means how much time a customer spends during the shopping process. Items column refers to how many items a customer buys while gender shows either the customer male or female. This training set is the input to classification algorithms which automatically analyze the input data and construct a model or classifier. The model could be a mathematical expression or a rule such as if then statement. The model in our case is a rule that if the per item minutes for any customer is greater or equal than 6 than the customer is female else a male i.e.

```
IF
        Items/Time >= 6
Then
        Gender= 'F'
else
        Gender = 'M'
```

The above rule is based on the common notion that females spend more time during shoping than male customers. Exceptions can be there and are treated as outliers.



NAME	Time	Items	Gender
Moin	10	2	M
Munir	16	3	M
Meher	15	1	F
Javed	5	1	M
Mahin	20	1	F
Akram	20	4	M

Figure-32.8: Classification model construction

Now once the model is built we have to test the accuracy of our classification model. For this purpose we take or test data set. Three randomly picked test data records have been shown in Figure 32.7. It is not the case that our test data contains only three records, here we have taken just three records for the sake of simplicity and let you understand the concept. We already know the classes of the test data set records i.e. we know their gender. The fact will be utilized to measure the accuracy of our model. So we will pick each of the three records one after another and put into the classifier. Since for the first record the ration is greater than 6 meaning that our model will assign it to the female class, but that may be an exception or noise. The second and the third records are as per rule. Thus, the accuracy of our model is 2/3 i.e. .66%. In other words we can say the confidence level of our classification model is 66%. The accuracy may change as we add more data. Now unseen data is brought into the picture. Suppose there is a record with name Firdous, time spent 15 minutes and 1 item purchased. We predict the gender by using our classification model and as per our model the customer is assigned 'F' (15/1=15 which is greater than 6). Thus we can easily predict the missing data with some reasonable accuracy using classification.

Clustering vs. Cluster Detection

In one-way clustering, reordering of rows (or columns) assembles clusters.

If the clusters are NOT assembled, they are very difficult to detect.

Once clusters are assembled, they can be detected automatically, using classical techniques such as K-means.

The difference between clustering and clustering is an important thing to understand. Basically these are two different concepts, mostly misperceived by novice data miners. First you cluster your data and then detect clusters in the clustered data. If you take unclustered input and try to find clusters, you may get some clusters or some part of a cluster no use of that. Thus until and unless clustering is performed, cluster detection is useless. The purpose here is to let you understand and remember the two mostly confusing concepts which are poles apart.

Clustering vs. Cluster Detection: EXAMPLE

Can you SEE the cluster in Fig-A?

How about Fig-B?

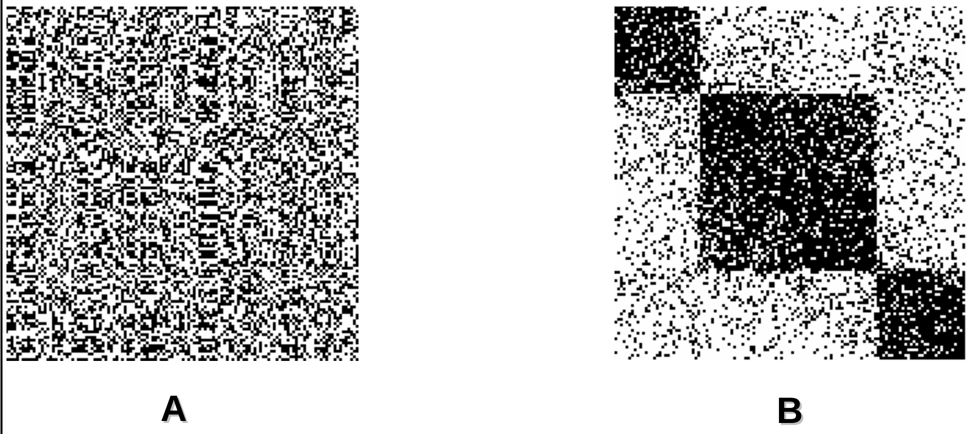


Figure-32.9: Clustering vs. Cluster Detection EXAMPLE

For better understanding consider the example in Figure 32.9. Can you tell the number of clusters, if there, by looking at Figure 32.9(A)? No you can't tell except that every point is a cluster but that is useless information. Cluster detection in Figure 32.9(A) is thus a very difficult task as clusters here are not visible. When clusters are detected, its not necessary that the clusters are visible as computer has no eyes. The clusters may be huge enough that you can not even display or even if displayed you can not visualize. So cluster visualization is not that much a problem. Now look at figure 32.9(B). The three clusters here are easily visible. We know that the Figures A and B are identical except that reordering has been performed on A in a scientific manner using our own indigenous technique. Figure 32.9(B) is the clustered output and when cluster detection will be performed on B, three clusters will successfully be detected. If A is taken as input to the cluster detection algorithm instead of B, you may end with nothing. There is a misconception among new data miners that cluster detection is a simple task. They think that using K-means is everything. This is not always the case, k-means can not work always . Can it work for matrices A and B? Wait till last slide for the answer.

The *K-Means* Clustering

- Given *k*, the *k-means* algorithm is implemented in 4 steps:
    - Partition objects into *k* nonempty subsets
    - Compute seed points as the centroids of the clusters of the current partition. The centroid is the center (mean point) of the cluster.
    - Assign each object to the cluster with the nearest seed point.
    - Go back to Step 2, stop when no more new assignment.

Before mentioning the strengths and weaknesses of the k-means, lets first discuss it working. It is implemented in four steps.

Step 1: In the first step you assign *k* clusters in your data set. Thus it's a supervised technique as you must know the number of classes and their properties a priori.

Step 2: The second step is to compute the seed points or centroids of your defined clusters i.e. which value is a most representative value of all the points in a cluster. For the sake of your convenience, we are talking about 2- D space, otherwise k-means can work for multidimensional data sets as well. The centroid can be the mean of these points, hence called k-means.

Step 3: In this step you take the distance of each point from the cluster centroids or means. On the basis of a predefined threshold value, it is decide that which point belongs to which cluster.

Step 4: You may repeat the above steps i.e. you find the means of newly formed clusters then find the distances of all points from those means and clusters are reconfigured. The process is normally repeated until some changes occur in clusters and mostly you get better results.

The *K-Means* Clustering: Example

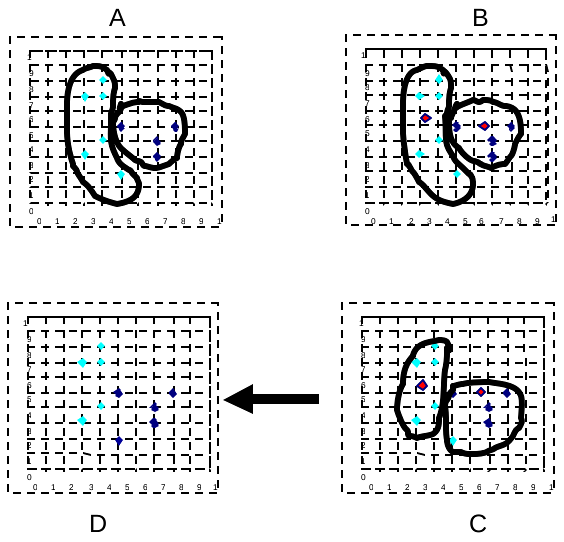


Figure-32.10: The *K-Means* Clustering Example

Consider the example in the figure 32.10 for better understanding k-means working. Figure A shows two sets of color points and the two colors represent two different classes. The polygons drawn around points in different clusters signify the cluster boundaries. Now at Figure B a red point has come in each of the two clusters. This is the centroid or mean of the value in that cluster. The next step is to measure the distances of all the points from each of these centoirds. So those distances which are above some threshold will go in a cluster for each mean point or centroid. Now look at the figure C, here on the basis of distances measured in the previous step new cluster boundaries have been made. In figure D the boundaries have been removed and we see that a point has been removed from one of the clusters and added to the other. As we will repeat the process, the result will get more and finer.

The *K-Means* Clustering: Comment

- Strength
  - *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t << n$.
  - Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*
- Weakness
  - Applicable only when *mean* is defined, then what about categorical data?
  - Need to specify *k,* the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

At the end of the lecture there are some comments about k-means:

1. K-means is a fairly fast technique and normally when terminates , then clusters formed are fairly good.
2. It can only work for data sets where there is the concept of mean (the answer to the question posed in a few slides back). If data is non numeric such as likes dislikes, gender, eyes color etc. then how to calculate means. So this is the first problem with the technique.

3. Another problem or limitation of the technique is that you have to specify the number of cluster in advance.

4. The third limitation is that it is not a robust technique as it not works well in presence of noise.

5. The last problem is that the clusters found by k-means have to be convex i.e. if you draw a polygon and join parameters of any two points in that polygon, that line goes out of the cluster boundaries.

Lay Out the Project

A data warehouse project is more like scientific research than anything in traditional IS!

The normal Information System (IS) approach emphasizes on knowing what the expected results are before committing to action. In scientific research, the results are unknown up front, and emphasis is placed on developing a rigorous, step-by-step process to uncover the truth. The activities involve regular interactions between the scientist and the subject and also among the project participants. It is advised to adopt an exploratory, hands-on process involving cross-disciplinary participation.

Building a data warehouse is a very challenging job because unlike software engineering it is quite a young discipline, and therefore, does not yet has well-established strategies and techniques for the development process. Majority of projects fail due to the complexity of the development process. To date there is no common strategy for the development of data warehouses; they are more of an art than science. Current data warehouse development methods can fall within three basic groups: data-driven, goal-driven and user-driven.

Implementation strategies
- Top down approach
- Bottom Up approach

Development methodologies
- Waterfall model
- Spiral model
- RAD Model
- Structured Methodology
- Data Driven
- Goal Driven
- User Driven

Implementation Strategies

Top Down & Bottom Up approach: A Top Down approach is generally useful for projects where the technology is mature and well understood, as well as where the business problems that must be solved are clear and well understood. A Bottom Up approach is useful, on the other hand, in making technology assessments and is a good technique for organizations that are not leading edge technology implementers. This approach is used when the business objectives that are to be met by the data warehouse are unclear, or when the current or proposed business process will be affected by the data warehouse.

Development Methodologies

A *Development Methodology* describes the expected evolution and management of the engineering system.

Waterfall Model: The model is a linear sequence of activities like requirements definition, system design, detailed design, integration and testing, and finally operations and maintenance. The model is used when the system requirements and objectives are known and clearly specified. While one can use the traditional waterfall approach to developing a data warehouse, there are several drawbacks. First and foremost, the project is likely to occur over an extended period of time, during which the users may not have had an opportunity to review what will be delivered. Second, in today's demanding competitive environment there is a need to produce results in a much shorter timeframe.

Spiral Model: The model is a sequence of waterfall models which corresponds to a risk oriented iterative enhancement, and recognizes that requirements are not always available and clear when the system is first implemented. Since designing and building a data warehouse is an iterative process, the spiral method is one of the development methodologies of choice.

RAD: Rapid Application Development (RAD) is an iterative model consisting of stages like scope, analyze, design, construct, test, implement, and review. It is much better suited to the development of a data warehouse because of its iterative nature and fast iterations. User requirements are sometimes difficult to establish because business analysts are too close to the existing infra-structure to easily envision the larger empowerment that data warehousing can offer. Development and delivery of early prototypes will drive future requirements as business users are given direct access to information and the ability to manipulate it. Management of expectations requires that the content of the data warehouse be clearly communicated for each iteration.

There are 5 keys to a successful rapid prototyping methodology:

1. Assemble a small, very bright team of database programmers, hardware technicians, designers, quality assurance technicians, documentation and decision support specialists, and a single manager.

2. Define and involve a small "focus group" consisting of users (both novice and experienced) and managers (both line and upper). These are the people who will provide the feedback necessary to drive the prototyping cycle. Listen to them carefully.

3. Generate a user's manual and user interface first. These will prove to be amazing in terms of user feedback and requirements specification.

4. Use tools specifically designed for rapid prototyping. Stay away from C, C++, COBOL, SQL, etc. Instead use the visual development tools included with the database.

5. Remember a prototype is NOT the final application. It servers a means of making the user more expressive about requirements and developing in them a clear understanding and vision of the system. Prototypes are meant to be copied into production models. Once the prototypes are successful, begin the development processing using development tools, such as C, C++, Java, SQL, etc.

Structured Development: When a project has more than 10 people involved or when multiple companies are performing the development, a more structured development management approach is required. Note that rapid prototyping can be a subset of the structured development approach. This approach applies a more disciplined approach to the data warehouse development. Documentation requirements are larger, quality control is critical, and the number of reviews increases. While some parts may seem like overkill at the time, they can save a project from problems, especially late in the development cycle.

Data-Driven Methodologies: Bill Inmon, the founder of data warehousing argues that data warehouse environments are data driven, in comparison to classical systems, which have a requirement driven development lifecycle. According to Inmon, requirements are the last thing to be considered in the decision support development lifecycle. Requirements are understood AFTER the data warehouse has been populated with data and results of queries have been analyzed by the end users. Thus the data warehouse development strategy is based on the analysis of the corporate data model and relevant transactions. This is an extreme approach ignoring the needs of data warehouse users a priori. Consequently company goals and user requirements are not reflected at all in the first cycle, and are integrated in the second cycle.

Goal-Driven Methodologies: In order to derive the initial data warehouse structure, Böhnlein and Ulbrich-vom Ende have presented a four-stage approach based on the SOM (Semantic Object Model) process modeling technique. The first stage determines goals and services the company provides to its customers. In the second step, the business process is analyzed by applying the SOM interaction schema that highlights the customers and their transactions with the process under study. In third step, sequences of transactions are transformed into sequences of existing dependencies that refer to information systems. The last step identifies measures and dimensions, by enforcing (information request) transactions, from existing dependencies. This approach is suitable only well when business processes are designed throughout the company and are combined with business goals.

Kimball also proposes a four-step approach where he starts to choose a business process, takes the grain of the process, and chooses dimensions and facts. He defines a business process as a major operational process in the organization that is supported by some kind of legacy system (or systems). We will discuss this in great detail in lectures 33-34.

User-Driven Methodologies: Westerman describes an approach that was developed at Wal-Mart and has its main focus on implementing business strategy. The methodology assumes that the company goal is the same for everyone and the entire company will therefore be pursuing the same direction. It is proposed to set up a first prototype based on the needs of the business. Business people define goals and gather, priorities as well as define business questions supporting these goals. Afterwards the business questions are prioritized and the most important business questions are defined in terms of data elements, including the definition of hierarchies. Although the Wal-Mart approach focuses on business needs, business goals that are defined by the organization are not taken into consideration at all.

Poe proposes a catalogue for conducting user interviews in order to collect end user requirements. She recommends int erviewing different user groups in order to get a complete understanding of the business. The questions should cover a very broad field including topics like job responsibilities.

WHERE DO YOU START?

The majority of successful data warehouses have started with a clear understanding of a business problem and the user requirements for information analysis. It is strongly recommended that the team assembled to create a data warehouse be comprised of IT professionals and business users. Projects must have a clearly defined scope for managing economic and operational limitations.

The process will be highly iterative as IT and end users work toward a reasonable aggregation level for data in the warehouse.

What specific Problems the DWH will solve?
Write down all the problems. The problems should be precise, clearly stated and testable i.e. success criteria is known or can easily be specified. Make sure to get user and management feedback by publicizing these written problems.

What criteria to use to measure success?
This is an often overlooked step in the problem definition. For every problem stated, you must define a means for determining the success of the solution. If you can't think of a success criterion, then the problem is not defined specifically enough. Stay away from problem statements such as "The data warehouse must hold all our accounting data." Restate the problem in quantifiable terms, like "The data warehouse must handle the current 20GB of accounting data including all metadata and replicated data with an expected 20% growth per year."

How to manage time and money?
The first data warehouse (first iteration's output) should cover a single subject area and be delivered at a relatively low cost. To minimize risk, the target platform should be one where IT has developed some infrastructure. Existing technical skills, operational skills and database experience will help tremendously. The project must be time boxed, with guaranteed deliverables every 90 days, and a project end date in six to nine months. The overall cost of the first data warehouse should be in the $200K to $500K range, with prototypes completed for $10K to $150K in 30 to 60 days (since local companies keep their costs secret, costs in dollar are given here as an example). Incremental successes will drive expansion of existing data warehouses and the funding and creation of additional ones.

What skills are required?
The level of complexity involved in successfully designing and implementing a data warehouse must not be underestimated. Time must be spent to acquire and develop additional skills for data warehousing developers and users. Some options are:

• Invest in just-in-time training (provided by data warehousing tool vendors)
• Use pilot projects as seeds for new technology training
• Develop reward systems that encourage experimentation
• Use outside system integrators and individual consultants

As additional motivation for data warehousing team members, a new class of job titles is being created. Companies are beginning to use dedicated titles such as: Data Warehouse Steward, Data Warehouse Architect, Data Quality Engineer and Data Warehouse Auditor.

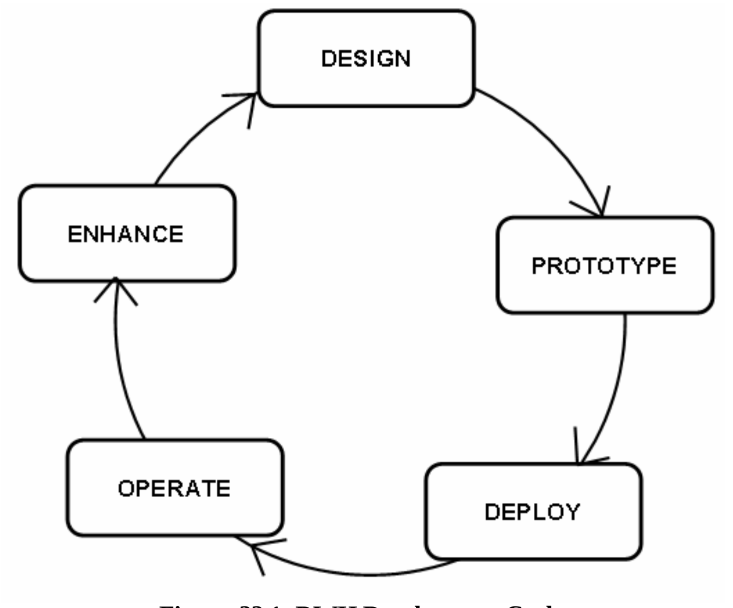


Figure-32.1: DWH Development Cycle

Although specific vocabularies vary from organization to organization, the data warehousing industry is in agreement of the fundamental data warehouse lifecycle model as shown in Figure 32.1. The cyclic model consists of 5 major steps described as follows

1. Design: It involves the development of robust star-schema-based dimensional data models from both available data and user requirements. It is thought that the best data warehousing practitioners even work with available organizational data and incompletely expressed user requirements. Key activities in the phase typically include end-user interview cycles, source system cataloguing, definition of key performance indicators and other critical business definitions, and logical and physical schema design tasks which feed the next phase of the model directly.

2. Prototype: In this step a working model of a data warehouse or data mart design, suitable for actual use, is deployed for a select group of end users. The prototyping purpose shifts, as the design team moves design-prototype-design sub-cycle. Primary objective is to constrain and /or reframe end-user requirements by showing them precisely what they had asked for in the previous iteration. As difference between stated needs and actual needs narrows down over iterations the prototyping shifts towards gaining commitment to the project at hand from opinion leaders in the end-user communities to the design, and soliciting their assistance in gaining similar commitment.

3. Deploy: The step includes traditional IT system deployment activities like formalization of user authenticated prototype for actual production use, document development, and training etc. Deployment involves two separate deployments (i) prototype deployment into a production –test environment (ii) Stress- and performance- tested production configuration deployment into an actual production environment. The phase also contains the most important and often neglected component of documentation. Lack of documentation may stall system operations as management people can not manage what they don’t know. Also, it may ultimately be used for educating the end users, prior to roll out.

4. Operation: The phase includes data warehouse/mart daily maintenance and management activities. The operations are performed to maintain data delivery services and access tools, and manage ETL processes that keep the data warehouse/mart current with respect to the authoritative source system.

5. Enhancement: The step involves modifications of physical technological components, operations and management processes (ETL etc.) and logical schema diagrams in response to changing business requirements. In situations of discontinuous changes, enhancement moves back into the fundamental design phase.

(Lecture based on "The data warehousing toolkit by Ralph Kimball and Margy Ross, 2nd Edition)

Business Dimensional Lifecycle: The Road Map Ralph Kimball's Approach
Implementing a data warehouse requires tightly integrated activities. As we discussed earlier, there are different DWH implementation strategies, we will be following Kimball's Approach. Kimball is considered as an authority in the DWH field, and his goal driven approach is a result of decades of practical experience. This presentation is a an overview of a data warehouse project lifecycle, based on this approach, from inception through ongoing maintenance, identifying best practices at each step, as well as potential vulnerabilities. It is believed that everyone on the project team, including the business analyst, architect, database designer, data stager, and analytic application developer, needs a high-level understanding of the complete lifecycle of a data warehouse.

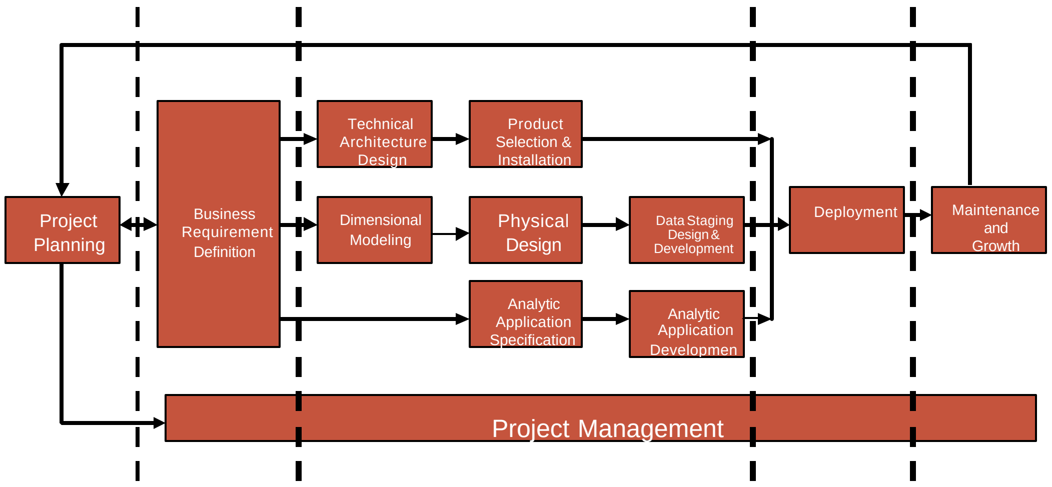

Project Planning
Business Requirement Definition
Technical Architecture Design
Product Selection & Installation
Dimensional Modeling
Physical Design
Data Staging Design & Development
Analytic Application Specification
Analytic Application Developmen
Deployment
Maintenance and Growth
Project Management

Figure -33.1: Business Dimensional Lifecycle (Kimball's Approach)

The business dimensional lifecycle framework, as shown in Figure 33.1, is depicted as a road map, that is extremely useful if we're about to embark on the unfamiliar journey of data warehousing. The Kimball's iterative data warehouse development approach drew on decades of experience to develop the business dimensional lifecycle. The name was because it reinforced several of key tenets for successful data warehousing. First and foremost, data warehouse projects must focus on the needs of the business. Second, the data presented to the business users must be dimensional. Finally while data warehousing is an ongoing process, each implementation project should have a finite cycle with a specific beginning and end. Ongoing project management serves as a foundation to keep the remainder of the lifecycle on track.

DWH Lifecycle: Key steps

1. **Project Planning**
2. **Business Requirements Definition**
3. **Parallel Tracks**
   - **3.1 Lifecycle Technology Track**
     - **3.1.1** Technical Architecture
     - 3.1.2 Product Selection
   - **3.2 Lifecycle Data Track**
     - **3.2.1** Dimensional Modeling
     - 3.2.2 Physical Design
     - 3.2.3 Data Staging design and development
   - **3.3 Lifecycle Analytic Applications Track**
     - 3.3.1 Analytic application specification
     - 3.3.2 Analytic application development
4. **Deployment**
5. **Maintenance**

Lifecycle Key Steps

Lifecycle begins with project planning during which we assess the organization's readiness for a data warehouse initiative, establish the preliminary scope and justification, obtain resources, and launch the project.

The second major task focuses on business requirements definition. The two -way arrow between project planning and business requirements definition (as shown in Figure 33.1) shows the much interplay between these two activities. Data warehouse designers must understand the needs of the business and translate them into design considerations. Business users and their requirements have an impact on almost every design and implementation decision made during the course of a warehouse project. In road map, this is reflected by the three parallel tracks that follow.

The top track deals with technology. Technical architecture design establishes the overall framework to support the integration of multiple technologies. Using the capabilities identified in the architecture des ign as a shopping list, we then evaluate and select specific products.

The middle track emanating from business requirements definition focuses on data. We begin by translating the requirements into a dimensional model which is then transformed into a physical structure. Physical design activities focus on performance tuning strategies, such as aggregation, indexing, and partitioning. Last but not least, data staging Extract-Transform-Load (ETL) processes are designed and developed.

The final set of tasks spawned by the business requirements definition is the design and development of analytic applications. The data warehouse project isn't done when we deliver data. Analytic applications, in the form of parameter-driven templates and analyses, will satisfy a large percentage of the analytic needs of business users.

Note:

1. Equally sized boxes (as shown in Figure 33.1) don't represent equally sized efforts, there is a vast difference in the time and effort required for each major activity

2. Data warehousing is an ongoing process, each implementation project should have a cycle with a specific beginning and an end.

DWH Lifecycle- *Step 1: Project Planning*

- Assessing Readiness
    - Factors
        - Business sponsor
        - Business motivation
        - Feasibility
        - Business/IT relationship
        - Culture
- Scoping

The DWH lifecycle begins with the project planning phase. It consists of multiple activities that must be performed before proceeding ahead in the lifecycle. Let's discuss the planning phase in detail;

Readiness and risk assessment: Before proceeding ahead with significant data warehouse expenditures, it is prudent to assess the organization's readiness to proceed. Five factors have been identified as leading indicators of data warehouse success; any shortfalls represent ris ks or vulnerabilities. Brief description in rank order of importance follows.

Business Sponsor: It is the most critical factor for successful data warehousing. Business sponsors should have a clear vision for the potential impact of a data warehouse on the organization. They should be passionate and personally convinced of the project's value while realistic at the same time. Optimally, the business sponsor has a track record of success with other intemal initiatives. He or she should be a politically astute leader who can convince his or her peers to support the warehouse.

Business motivation: The second readiness factor is having a strong, compelling business motivation for building a data warehouse. This factor often goes hand in hand with sponsorship. A data warehouse project can't merely deliver a nice-to-have capability; it needs to solve critical business problems in order to garner the resources required for a successful launch and healthy lifespan.

Feasibility: There are several aspects of feasibility, such as technical or resource feasibility, but data feasibility is the most crucial. Are we collecting real data in real operational source systems to support the business requirements? Data feasibility is a major concern because there is no short-term fix if we're not already collecting reasonably clean source data at the right granularity.

Business/IT relationship: The fourth factor focuses on the relationship between the business and IT organizations. In your company, does the IT organization understand and respect the business? Conversely, does the business understand and respect the IT organization? The inability to honestly answer yes to these questions doesn't mean that you can't proceed. Rather, the data warehouse initiative can be an opportunity to mend the fence between these organizations, assuming that both deliver.

Culture: The final aspect of readiness is the **current analytic culture** within your company. Do business analysts make decisions based on facts and figures, or are their decisions based on intuition, anecdotal evidence, expert judgment or gut feeling?

Scoping: Requires the joint input of both the IT organization and business management. The scope should be both meaningful in terms of its value to the organization and manageable. When you are first getting started, you should focus on data from a single business process. Save the more challenging, cross-process projects for a later phase. Sometimes project teams feel that the delivery schedule is cast in concrete before project planning is even initiated. The prioritization process can be used to convince IT and business management that adjustments are required. Finally, remember to avoid the *law of too* when scoping-too firm of a commitment to too brief of a timeline involving too many source systems and too many users in too many locations with too diverse analytic requirements.

DWH Lifecycle- *Step 1: Project Planning*

- Justification (cost vs. benefit)
- Team development
- Project Plan
  - Identifying all tasks.
  - User acceptance, milestones and deliverables.
  - Making and following a communication matrix.
  - Avoiding scope creep.
  - Partnership with business user.
- Keys to project planning & Management
  - Business sponsor
  - Scope
  - Best team
  - Excellent project manager

Justification requires an estimation of the benefits and costs associated with a data warehouse. The anticipated benefits grossly outweigh the costs. IT usually is responsible for deriving the expenses. You need to determine approximate costs for the requisite hardware and software. Data warehouses tend to expand rapidly, so be sure the estimates allow some room for short-term growth.

Team Development (Staffing): Data warehouse projects require the integration of a cross functional team with resources from both the business and IT communities. From the business side of the house, the roles needed are business sponsor, business driver, business lead and business users. Several other positions are staffed from either the business or IT organizations. These are business system analyst, business subject matter expert, analytic application developer and data warehouse educator. The roles typically staffed from the IT organization (or an external consulting firm) are project manager, technical architect, technical support specialists, data modeler, database administrator, metadata coordinator, data steward, data staging designer, data staging developer, and data warehouse support.

Establishing the core team

Hardest to find talent.: *The talent you need on the core team is the hardest to find*

1. Part-time: Data modeler and database analyst (DBA)

2. **Full-time:** Experienced & educated people with least one successful implementation.

3. **Full-time:** (i) DSS Data analyst (ii) 4GL programmer with information center or end-user support background (iii) experienced data–centric developer.

- Data engineers to get started and create potential.
- Data analysts or data usage specialists to stay in business.

So, who should be on this small core team? Typically, the first talent hired is a data modeler and the second is a database analyst or DBA. While you need these data specialists, you don't need them as fulltime, permanent participants of the core team. The best individuals at the core of your team are people who intimately understand the broad issues of a complete data warehouse program. Ideally, they studied the field extensively and participated in at least one successful project. These folks are hard to come by.

Your next choice should be those with experience of packaging data for consumer use: a data analyst with DSS experience, a 4GL programmer with information center or end-user support background, a data–centric developer with real experience using spiral or interactive methods. You need data engineers to get started and to add new data resources to the warehouse. They create the potential. You will need data analysts or data usage specialists to stay in business. They deliver results.

Project Plan: Developing the data warehouse project plan involves identification of all the tasks necessary to implement the data warehouse. The project plan should identify a user acceptance checkpoint after every major milestone and deliverable to ensure that the project is still on track and that the business is still intimately involved. The data warehouse project demands broad communication. During the project planning phase a communication matrix is helpful to make certain that the communication strategy is executed. Data warehouse projects are vulnerable to scope creep largely due to our strong desire to satisfy users' requirements. We need to be most watchful about the accumulation of minor changes that snowball. The most important thing to remember about scope changes is that they shouldn't be made in an IT vacuum. The right answer depends on the situation. Now is the time to leverage your partnership with the business to arrive at an answer with which everyone can live.

The keys to data warehouse project planning and management include:

1. Having a solid business sponsor
2. Balancing high value and doability to define the scope
3. Working with the best team possible to develop a detailed project plan
4. Being an excellent project manager by motivating, managing, and communicating up, down, and across the organization.

DWH Lifecycle- *Step: 2 Requirements Definition*

- Requirements preplanning
- Requirements collection
- Post collection

The second phase in the DWH lifecycle is requirements definition. Performing a requirements analysis is critical to the success of any project. W ithout a clear goal in mind, success is dubious. Establishing a broad view of the business' requirements should always be the first step. The understanding gained will guide everything that follows, and the details can be filled in for each phase in turn. The double headed arrow between the planning and requirements definition phase indicates that user requirements drives not only the succeeding phases but the preceding phase as wee. The changes in user requirements may effect the project plan. This phase is accomplished in three steps

Requirements preplanning: This phase consists of activities like choosing the forum, identifying and preparing the requirements team and finally selecting, scheduling and preparing the business representatives.

Business requirements collection: The requirements collection process flows from an introduction through structured questioning to a final wrap-up. The major activities involved are, launching, determining interview flow, wrapping up and conducting data centric interviews.

Post collection: The phase consists of steps like debriefing, documentation, prioritization and consensus**.**

Each of the phases is discussed in detail in the following slides

DWH Lifecycle- *Step: 2 Requirements Definition*

Requirements Preplanning

- Forum
    - Interviews
    - Facilitated sessions
    - Hybrid

- Requirements team

- Business representatives
  - Selecting
  - Scheduling
  - Preparing

Requirements Preplanning
Before sitting down with the business community to gather requirements, it is suggested to set you up for a productive session by considering the following:

Choose the Forum: There are two primary techniques for gathering requirements i.e. interviews or facilitated sessions. Both have their advantages and disadvantages. Interviews encourage lots of individual participation. They are also easier to schedule. Facilitated sessions may reduce the elapsed time to gather requirements, although they require more time commitment from each participant. Kimball prefers using a hybrid approach with interviews to gather the gory details and then facilitation to bring the group to consensus. However, the forum choice depends on the team's skills, the organization's culture, and what you've already subjected your users to. This is a case in which one size definitely does not fit all.

Identify and Prepare the Requirements Team: Regardless of the approach, you need to identify and prepare the project team members who are involved. If you're doing interviews, you need to identify a lead interviewer whose primary responsibility is to ask the great open-ended questions. Meanwhile, the interview scribe takes copious notes. Before you sit down with users, you need to make sure you're approaching the sessions with the right mindset. Since the key to getting the right answers is asking the right questions, we recommend that questionnaires be formulated before user meetings. It is a tool to organize your thoughts and serve as a fallback device in case your mind goes blank during the interview session.

Business Representatives Soliciting: If this is first foray into data warehousing, talk to business people that represent horizontal breadth across the organization. This coverage is critical to formulating the data warehouse bus matrix blueprint. Within the target user community, one should cover the organization vertically.

Scheduling: Schedule representatives nicely. The scheduler needs to allow ½ hour between meetings for debriefing and other necessities. Interviewing is extremely taxing because you must be completely focused for the duration of the session. Consequently, it is recommended to schedule three to four sessions in a day because the interviewers get very tired after that, and productivity goes down.

Preparing: The optimal approach is to conduct a project launch meeting with the users. The launch meeting disseminates a consistent message about the project. The interview team must prepare the interviewees by highlighting the topics to be covered in the upcoming session. It is advised that do not include a copy of the questionnaire, which is not intended for public dissemination. One can ask the interviewees to bring copies of their key reports and analyses.

DWH Lifecycle- ***Step : 2 Requirements Definition***

Requirements Collection

- Launch
- Interview flow
    - Rules for effective interviewing
        - Role
        - Flow
        - Consistency
        - Vocabulary
- Wrap up
- Conducting data centric interviews

Launch: Responsibility for introducing the interview should be established prior to gathering in a conference room. The designated kickoff person should script the primary points to be conveyed in the first couple of minutes. The introduction should convey a crisp, business-centric message rather than rambling on about the hardware, software, and other technical jargon.

Interview Flow: The objective of an interview is to get business users to talk about what they do, and why they do it. Determining how they track progress and success translates directly into the dimensional model. If we're meeting with a person who has more hands-on data experience, we indirectly probe to better understand the dimensionality of the business, along with hierarchical roll-ups. If the interviewee is more analytic, we ask about the types of analyses currently being performed. If we are meeting with business executives, we usually don't delve into the details, and just ask about their vision for better leveraging information in the organization.

***Rules for Effective interviewing include*:** (i) Remember your interview role and listen and absorb like a sponge. (ii) Strive for a conversational flow, don't dive too quickly. (iii) Verify communication and capture terminology precisely because mostly its inconsistent (iv) establish a peer basis with the interviewee; use his or her vocabulary.

Wrap-up: At the conclusion of interview, ask each interviewee about his success criteria for the project which should be measurable. At this point, the interviewees must understand that just because we discussed a capability in the meeting doesn't guarantee that it'll be included in the first phase of the project. We thank interviewees for their brilliant insights and let them know what's happening next and what their involvement will be.

Conducting Data-Centric Interviews: While we're focused on understanding the requirements of the business, it is helpful to intersperse sessions with the source system data gurus or subject matter experts to evaluate the feasibility of supporting the business needs. These data-focused interviews are quite different from the ones just described. The goal is to assess that the necessary core data exists before momentum builds behind the requirements. A more complete data audit will occur during the dimensional modeling process. We're trying to learn enough at this point to manage the organization's expectations appropriately.

DWH Lifecycle- *Step : 2 Requirements Definition*

- **Post Collection**
    - Debriefing
        - Same "frequency"

    - Fill-in gaps
    - Review reports/material

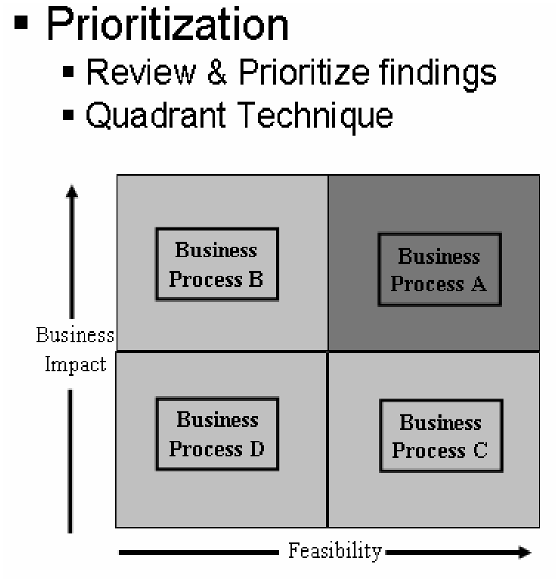
- Documentation
    - Meaningful write-up of each individual interview.
    - Identify key business process
    - Requirements of key processes
    - Process x group/user matrix

Debriefing: Immediately following the interview, the interview team should debrief. You want to ensure that you're on the same page about what was learned, as well as being prepared for any surprises or inconsistencies. It is also helpful to review your notes quickly to fill in any gaps while the interview is still fresh in your memory. Likewise, you should examine the reports gathered to gain further offline insight into the dimensionality that must be supported in the data warehouse.

Documentation: While documentation is everyone's least favorite activity, it is critical for both user validation and project team reference materials. There are two levels of documentation that typically result from the requirements process. The first is to write up each *individual* interview. This activity can be quite time-consuming because the write-up should not be merely a stream-of-consciousness transcript but should make sense to someone who wasn't in the interview. The second level of documentation is a consolidated findings document. Organize the document by first identifying the key business processes. Consequently, it is logical to organize the requirements of the business into the same buckets that will, in turn, become implementation efforts. Sometimes the processes are brought together in a matrix to convey the opportunities across the organization. The rows of the opportunity matrix identify the business processes while the columns identify the organizational groups or functions.

Prioritization and Consensus: ***Four Cell Quadrant Technique***

The requirements findings document serves as the basis for presentations back to senior management representatives. The requirements wrap-up presentation is positioned as a findings review and prioritization meeting. Once the findings have been reviewed, it is time to prioritize. The four-cell quadrant technique (shown in Figure 33.2) is an effective tool for reaching consensus on a data warehouse development plan that focuses on the right initial opportunities. The quadrant's vertical axis refers to the potential impact or value to the business. The horizontal axis conveys feasibility of each of the findings.



- Prioritization
  - Review & Prioritize findings
  - Quadrant Technique

Figure -33.2: The quadrant method

So the process having higher feasibility and impact is given higher priority over the process having lower feasibility and impact. In example of Figure 33.2, process A has highest priority while the process D has lowest priority.

(Lecture based on "The data warehousing toolkit by Ralph Kimball and Margy Ross, 2nd Edition)

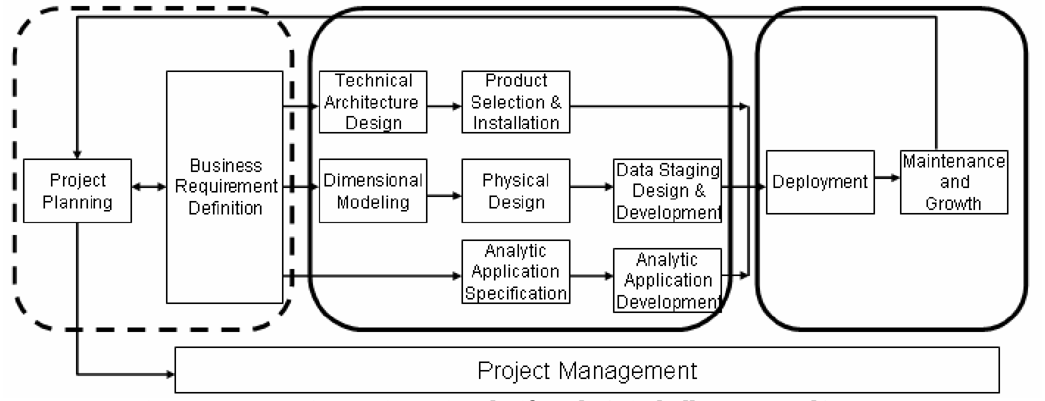

Project Planning
Business Requirement Definition
Technical Architecture Design
Product Selection & Installation
Dimensional Modeling
Physical Design
Data Staging Design & Development
Analytic Application Specification
Analytic Application Development
Deployment
Maintenance and Growth
Project Management

Figure: 34.1: Business Dimensional Lifecycle (Kimball's Approach)

We have already discussed that the DWH development lifecycle (Kimball's Approach) has three parallel tracks emanating from requirements definition. These are technology track, data track and analytic applications track. The focus will be on technology track and analytic track as we have discussed thoroughly about data in previous lectures. So first of all we will discuss technology track in detail.

Technical Architecture Design

Constructing a kitchen involves mason, plumber (water, gas), carpenter, electrician, iron-smith, painter, interior designer AND Takhedar.

Need a common document which links the works of all, i.e. blue-print or a map NEED Architect.

Tech. Arch. design is a similar document.

Help catch problems on paper and minimize surprises.

As shown in Figure34.1, the first task in technology track is the technical architecture design. What is meant by architecture? Is it really needed? The questions can be answered by considering the following analogy. Before constructing a home, we must have a complete home designed by some good architect. To further simplify things, let's narrow down our example to a single unit of home like a kitchen. Building a kitchen involves people like mason, plumber both for gas and water, carpenter, electrician and the list goes on. All have their own expertise and specific tasks to perform which when integrate result in a fully functional kitchen. But how these roles coordinate? where should be the sink, where should be the entrance, where should be the window (if there), where should be the cupboards etc. All these are specified by the architect who is the person who develops and gives the overall construction plan i.e. architecture of the kitchen keeping in view the overall home.

The architecture plan thus provides an overall picture and a blue print of all facilities in combine. It's a document that is used by the contractor (implementer) for scheduling different construction activities.

The architecture plan, in both cases, allows us to catch problems on paper (such as having the dishwasher too far from the s ink) and minimize mid project surprises. It supports the coordination of parallel efforts while speeding development through the reuse of modular components.

Technical Architecture Design: More

Identify components needed i.e. now versus required at a later stage sink vs. gas-light.

Organizing framework for integration of technologies

Supports communication regarding a consistent set of technical requirements:

- Within the team
- Upward to management, and
- Outward to vendors.

Since architecture is a document that gives an overall picture, the major components are easily identifiable. Similarly we can also identify immediately required components versus those which will be incorporated at a later stage. In our kitchen analogy the sink is a must component, a kitchen without a sink is not a kitchen. Similarly, having a gas-light in kitchen is a nice-to-have component. It is better if some kitchen has gas-light for electric load shading days but a kitchen without it, will still be a kitchen.

Thus much like a blue print for a new home, the technical architecture is the blueprint for the warehouse's technical services and elements. The architecture plan serves as an organizing framework to support the integration of technologies.

Most importantly, the architecture plan serves as a communication tool. Kitchen construction blueprints allow the architect, general contractor, subcontractors, and homeowner to communicate from a common document. The plumber knows that the electrician has power in place for the garbage disposal. Likewise, the data warehouse technical architecture supports communication regarding a consistent set of technical requirements within the team, upward to management, and outward to vendors. It provides a common ground for all stakeholders for mutual and consistent understanding of the overall system.

Note: All points discussed or to be discussed may not be completely applicable in our local environment

DWH Lifecycle- *Step 3.1:Technology Track*

8-Step Process

1. Establish an Architecture Task Force (2-3 people)

2. Collect Architecture-Related Requirements
   - Business needs => HW not other way round
   - Architectural implications of business needs
   - Timing, performance and availability needs
   - Talk to IT people for current standards, directions and boundaries.

- Lessons learned.

3. Document Architecture Requirements
   - Make a matrix (row business process & column architectural implication)
   - Global sales coverage => 24/7 availability, data mirroring, adequate network bandwidth etc.

8 Step Process
Data warehouse teams approach the technical architecture design process from opposite ends of the spectrum. Some teams are so focused on data warehouse delivery that the architecture feels like a distraction and impediment to progress and eventually, these teams often end up rebuilding. At the other extreme, some teams want to invest two years designing the architecture while forgetting that the primary purpose of a data warehouse is to solve business problems, not address any plausible (and not so plausible) technical challenge. Neither end of the architecture spectrum is healthy; the most appropriate response lies somewhere in the middle. Kimball suggests an eight-step process for building technical architecture. All steps will be discussed in detail one by one.

1. **Establish an Architecture Task Force**

It is most useful to have a small task force of two to three people focus on architecture design. Typically these are technical architect, the data staging designer and analytic application developer. This group needs to establish its charter and deliverables time line. It also needs to educate the rest of the team (and perhaps others in the IT organization) about the importance of architecture.

2. **Collect Architecture-Related Requirements:**

Defining the technical architecture is not the first box in the lifecycle diagram, as shown in Figure 34.1. This implies that the architecture is created to support high value business needs; it's not meant to be an excuse to purchase the latest, greatest products.

The key input into the design process should come from the business requirements definition findings with a slightly different filter to drive the architecture design. The focus is to uncover the architectural implications associated with the business's critical needs e.g. like any timing, availability, and performance needs.

In addition to leveraging the business requirements definition process, additional interviews within the IT organization are also conducted. These are purely technology-focused sessions to understand current standards, planned technical directions, and nonnegotiable boundaries.

Also, lessons learned from prior information delivery projects, as well as the organization's willingness to accommodate operational change on behalf of the warehouse, can be uncovered such as identifying updated transactions in the source system.

3. **Document Architecture Requirements**

Once the business requirements definition process is leveraged and supplemental IT interviews conducted, the findings need to be documented. A simplistic MATRIX can be used for this purpose. The rows of the matrix list each business requirement that has an impact on the architecture, while matrix columns contain the list of architectural implications.

As an example supposes that a business is spread globally and there is a need to deliver global sales performance data on a nightly basis. The technical implications might include 24/7 worldwide availability, data mirroring for loads, robust metadata for support global access, adequate network bandwidth, and sufficient staging horsepower to handle the complex integration of operational data and so on.

DWH Lifecycle- *Step 3.1:Technology Track*

4. Develop a high-level Arch. Model
   - Several days of heavy thinking in conference room.
   - Grouping of requirements (data staging, data access, meta)
   - High level refinement (up-front, nuts-bolts hidden) of major systems.

5. Design and Specify the Subsystems
   - For each subsystem (data staging), detailed list of capabilities.
   - Do research (internet, peers etc.) more graphic models generated.
   - Also consider security, physical infrastructure, and configuration.
   - Sometimes infrastructure i.e. HW and SW pre-determined.
   - Determine Architecture Implementation Phases.
   - For more than 1TB DWH, revisit infrastructure.

After the architecture requirements have been documented, models are formulated to support the identified needs the architecture task force often sequesters itself in a conference room for several days of heavy thinking. The team groups the architecture requirements into major components, such as data staging, data access, metadata, and infrastructure. From there the team drafts and refines the high-level architectural model. This drawing is similar to the front elevation page on housing blueprints. It illustrates what the warehouse architecture will look like from the street, but it is dangerously simplistic because significant details are embedded in the pages that follow.

It is time now to do a detailed design of the subsystems. For each component, such as data staging services, the task force will document a laundry list of requisite capabilities. The more specific, the better, because what's important to your data warehouse is not necessarily critical to mine. This effort often requires preliminary research to better understand the market. Fortunately, there is no shortage of information and resources available on the Internet, as well as from networking with peers. The subsystem specification results in additional detailed graphic models. In addition to documenting the capabilities of the primary subsystems, we also must consider our security requirements, as well as the physical infrastructure and configuration needs. Often, we can leverage enterprise-level resources to assist with the security strategy. In some cases the infrastructure choices, such as the server hardware and database software, are predetermined. At this point in time we must have got an idea of what should be the implementation steps/phases that will be used for the DWH implementation. However, if building a large data warehouse, over 1 TB in size, we should revisit these infrastructure platform decisions to ensure that they can scale as required. Size, scalability, performance, and flexibility are also key factors to consider when determining the role of OLAP cubes in the overall technical architecture.

DWH Lifecycle- *Step 3.1:Technology Track*

6. Determine Architectural implementation phases

- Can't implement everything simultaneously.
- Some are negotiable mandatory, others nice-to-haves, later.
- Business requirements set the priority.
- Priorities assigned by looking at all the requirements.

7. Document the Technical Architecture

- Document phases decided.
- Material for those not present in the conference room.
- Adequate details for skilled professional (carpenter in kitchen)

8. Review and finalize the technical architecture

- Educate organization, manage expectations.
- Communicate to varying level of details to different levels of team
- Subsequently, put to use immediately for product selection

Like the Kitchen's analogy, we likely can't implement all aspects of the technical architecture at once. Some are nonnegotiable mandatory capabilities, whereas others are nice-to-haves that can be deferred until a later date. Again, we refer back to the business requirements to establish architecture priorities. Business requirements drive the architecture and not the other way round. We must provide sufficient elements of the architecture to support the end-to-end requirements of the initial project iteration. It would be ineffective to focus solely on data staging services while ignoring the capabilities required for metadata and access services.

We need to document the technical architecture, including the planned implementation phases, for those who were not sequestered in the conference room. The technical architecture plan document should include adequate detail so that skilled professionals can proceed with construction of the framework, much like carpenters frame a house based on the blueprint. Eventually the architecture building process (Technology Track) comes to an end.

With a draft plan in hand, the architecture task force is back to educating the organization and managing expectations. The architecture plan should be communicated, at varying levels of detail, to the project team, IT colleagues, business sponsors, and business leads. Following the review, documentation should be updated and put to use immediately in the product selection process.

DWH Lifecycle- *Step 3.1:Technology Track*

3.1.2 Product selection and Installation

- Understand corporate purchasing process

- Product evaluation matrix
  - Not too vague/generic
  - Not too specific

- Market research (own ugly son)
  - Understand players and offerings
  - Internet, colleagues, exhibitions etc.
  - RFP is an option, but time consuming and beauty contest

- Narrow options, perform detailed evaluations
  - Few vendors can meet tech.& functional requirements
  - Involve business reps.
  - You drive the process, not the vendors.
  - Centered around needs, not bells -and-whistles.
  - Talk to references of similar size installations.

Understand the corporate purchasing process: The first step before selecting new products is to understand the internal hardware and software purchase approval processes, whether we like them or not. Perhaps expenditures need to be approved by the capital appropriations committee. Or you may be asked to provide a bank guarantee against the funds released to buy hardware.

Develop a product evaluation matrix: Using the architecture plan as a starting point, we develop a spreadsheet-based evaluation matrix that identifies the evaluation criteria, along with weighting factors to indicate importance. The more specific the criteria, the better. If the criteria are too vague or generic, every vendor will say it can satisfy our needs. On the other hand, if the criterion is too specific, everyone will shout favoritism.

Conduct market research: We must be informed buyers when selecting products, which mean more extensive market research to better understand the players and their offerings. We must not place the ball in vendor's court because he will never bring forth limitations of his tool. Its like once a *Badsha Salamat* asked his *Wazir* to bring the most beautiful child of his Kingdom. *Wazir* returned thrice with the same boy who was ugly. *Badshah* warned his *Wazir* of severe consequences and gave him yet another chance to search. *Wazir* returned with the same boy again. *Badshah* was astonished and angry (at the same time) and asked his *Wazir* why he was bringing the same ugly boy again and again although he was not up to the standards? *Wazir* replied that he had walked around all over the town, but couldn't find anyone as beautiful as that boy, who was his son. Thus we must not rely on vendors and must make self efforts to gain as much insight into the tools as possible. For this purpose, we can use potential research sources including the Internet, industry publications, colleagues, conferences, vendors, exhibitions and analysts (although be aware that analyst opinions may not be as objective as we're lead to believe).

Narrow options to a short list and perform detailed evaluations. Despite the plethora of products available in the market, usually only a small number of vendors can meet both our functionality and technical requirements. By comparing preliminary scores from the evaluation matrix, we should focus on a narrow list of vendors about whom we are serious and disqualify the rest. Once we're dealing with a limited number of vendors, we can begin the detailed evaluations. Business representatives should be involved in this process if we're evaluating data access tools. As evaluators, we should drive the process rather than allow the vendors to do the driving. We share relevant information from the architecture plan so that the sessions focus on our needs rather than on product bells and whistles. Be sure to talk with vendor references, both those provided formally and those elicited from your informal network. If possible, the references should represent similarly sized installations.

DWH Lifecycle- *Step 3.1:Technology Track*

3.1.2 Product selection and Installation

- Conduct prototype, if necessary
  - If one clear winner bubbles up, it is good.
  - Winner due to experience, relationship, commitment
  - Prototype with no more than two products
  - Demonstrate using a limited, yet realistic application using flat text file.

- Keep the competition "hot"
  - Even if single winner, keep at least two in
  - Use virtual competition to bargain with the winner

- Select product, install on trial, and negotiate
  - Make private not public commitment.
  - Don't let the vendor you are completely sold.
  - During *trial period*, put to real use.
  - Near the end of trial, negotiate.

Conduct prototype, if necessary: After performing the detailed evaluations, sometimes a clear winner bubbles to the top, often based on the team's prior experience or relationships. In other cases, the leader emerges due to existing corporate commitments. In either case, when a sole candidate emerges as the winner, we can bypass the prototype step. If no vendor is the apparent winner, we conduct a prototype with no more than two products.

Keep the competition "hot": Even if a single winner is left, it is a good piece of advice that always keep at least two. What if you keep one? The sole vendor may take benefit of the situation that he is the only player and create a situation favorable for him. He might get an upper hand in the bargaining process, and mold things according to his facility and benefit. To avoid such a situation enlist a competitor too, even if a single vendor is the winner. This will create a competitive environment which may ultimately turn into your favor.

Select product, install on trial, and negotiate: It is time to select a product. Rather than immediately signin g on the dotted line, preserve your negotiating power by making a private, not public, commitment to a single vendor. Embark on a *trial period* where you have the opportunity to put the product to real use in your environment. As the trial draws to a close, you have the opportunity to negotiate a purchase that's beneficial to all parties involved.

DWH Lifecycle- *Step 3.3: Analytic Applications Track*

- **Overview**
  - Design and develop applications for analysis.
  - It is really the "fun part".
  - Technology used to help the business.
  - Strengthen relationship between IT and business user.
  - The DWH "face" to the business user.
  - Querying NOT completely ad-hoc.
  - Parameter driven querying satisfy large % of needs.
  - Develop consist analytic frame-work instead of shades of Excel macros.

The final set of parallel activities following the business requirements definition in Figure 34.1 is the analytic application track, where we design and develop the applications that address a portion of the users' analytic requirements. As a well-respected application developer once told, "Remember, this is the fun part!" We're finally using the investment in technology and data to help users make better decisions. The applications provide a key mechanism for strengthening the relationship between the project team and the business community. They serve to present the data warehouse's face to its business users, and they bring the business needs back into the team of application developers.

While some may feel that the data warehouse should be a completely ad hoc query environment, delivering parameter-driven analytic applications will satisfy a large percentage of the business community's needs. There's no sense making every user start from scratch. Constructing a set of analytic applications establishes a consistent analytic framework for the organization rather than allowing each Excel macro to tell a slightly different story. Analytic applications also serve to encapsulate the analytic expertise of the organization, providing a jump-start for the less analytically inclined.

DWH Lifecycle- *Step 3.3: Analytic Applications Track*

- **3.3.1 Analytic applications specification**
  - Starter set of 10-15 applications.
  - Prioritize and narrow to critical capabilities.
  - Single template use to get 15 applications.
  - Set standards: Menu, O/P, look feel.
  - From standard: Template, layout, I/P variables, calculations.
  - Common understanding between business & IT users.

Following the business requirements definition, we need to review the findings and collected sample reports to identify a starter set of approximately 10 to 15 analytic applications. We want to narrow our initial focus to the most critical capabilities so that we can manage expectations and ensure on-time delivery. Business community input will be critical to this prioritization process. While 15 applications may not sound like much, the number of specific analyses that can be created from a single template merely by changing variables will surprise you.

Before we start designing the initial applications, it's helpful to establish standards for the applications, such as common pull-down menus and consistent output look and feel. Using the standards, we specify each application template, capturing sufficient information about the layout, input variables, calculations, and breaks so that both the application developer and business representatives share a common understanding.

During the application specification activity, we also must give consideration to the organization of the applications. We need to identify structured navigational paths to access the applications, reflecting the way users think about their business. Leveraging the Web and customizable information portals are the dominant strategies for disseminating application access.

DWH Lifecycle- *Step 3.3: Analytic Applications Track*

- **3.3.2 Analytic applications development**
  - Standards: naming, coding, libraries etc.
  - Coding begins AFTER DB design complete, data access tools installed, subset of historical data loaded.
  - Tools: Product specific high performance tricks, invest in tool-specific education.
  - Benefits: Quality problems will be found with tool usage => staging.
  - Actual performance and time gauged.

When we move into the development phase for the analytic applications, we again need to focus on standards. Standards for naming conventions, calculations, libraries, and coding should be established to minimize future rework. The application development activity can begin once the database design is complete, the data access tools and metadata are installed, and a subset of historical data has been loaded. The application template specifications should be revisited to account for the inevitable changes to the data model since the specifications were completed.

Each tool on the market has product-specific tricks that can cause it to literally walk on its head with eyes closed. Therefore, rather than trying to learn the techniques via trial and error, you should invest in appropriate tool-specific education or supplemental resources for the development team.

While the applications are being developed, several ancillary benefits result. Application developers, armed with a robust data access tool, quickly will find needling problems in the data haystack despite the quality assurance performed by the staging application. This is one reason why we prefer to get started on the application development activity prior to the supposed completion of staging. Of course, we need to allow time in the schedule to address any flaws identified by the analytic applications. The developers also will be the first to realistically test query response times. Now is the time to begin reviewing our performance-tuning strategies.

The application development quality-assurance activities cannot be completed until the data is stabilized. We need to make sure that there is adequate time in the schedule beyond the final data staging cutoff to allow for an orderly wrap-up of the application development tasks.

DW Lifecycle - Step 4: Deployment

- The three tracks converge at deployment.
- Not natural, require substantial pre-planning, courage, will-power and honesty.
- Something like waiting for arrival of *Baarat* , as only then meals can be served
- The *Dulaaha* is the key like data.
- Should serve uncooked data i.e. *Dulaaha* under a *Sehara* (not possible now) or wait and miss the deadline.

The technology, data, and analytic application tracks converge at deployment. Deployment is similar to serving meal at a large marriage gathering while the *Baraat* is late. The *Baraat* has not reached yet, only then the meal will be served. The organizers from *Dulhans* side are in a fix. If they serve meal without *Dulha,* there can be a big showdown, and *Dulha* may decide to go back. If they don't serve the meals, guests are getting angry and hungry by the minute. It can be difficult to predict exactly how long it will take to serve the food, as it is linked with the arrival of the *Baraat*. If it were old days when *Dulhas* used to wear *sehras* it was quite possible to place anybody wearing *sehra* and serve food instead of waiting and missing the deadline, but that's no more a norm nowadays. In the case of data warehouse deployment, the data (is the main entree, analogous to the *Dulha*). Unfortunately, in data warehousing, even if the data isn't fully available or ready, some people often still proceed with deployment because we have told the warehouse users that they'd be served on a specific date and time (i.e. when the *Dulha* and his relatives will arrive).

DW Lifecycle - Step 4: Deployment

- **Other than data readiness, education and support are critical.**

- **Educate on complete warehouse deliverable:**
  - Data
  - Analytic applications
  - Data access tools
  - Education tools (1 to 2 days of development for each hour of education)

- **For effective education:**
  - Understand the audience, don't overwhelm.
  - Train after delivery of data and analytic applications
  - Postpone education, if DWH not ready.
  - "No education, no access policy".

Readiness assessment: Perhaps more important, a successful deployment demands the courage and willpower to assess the project's preparedness to deploy honestly. Deployment is similar to serving meal to friends and relatives on a wedding, which we have already discussed. It can be difficult to predict exactly how long it will take for the *Baraat* to arrive.

Education: Since the user community must accept the warehouse for it to be deemed successful, education is critical. The education program needs to focus on the complete warehouse deliverable: data, analytic applications, and the data access tool (as appropriate). Consider the following for an effective education program; **(i)** ***Understand your target audience***, don't overwhelm, ***(ii) Don't train*** the business community early prior to the availability of data and analytic applications, ***(iii) No release, no education:*** Postpone the education (and deployment) if the data warehouse is not ready to be released, (iv) Gain the sponsor's commitment to a “no ***education, no access***" policy.

DW Lifecycle- Step 5: Maintenance and Growth

- Support
- Education
- Technical support
- Program support
- Growth

We've made it through deployment, so now we're ready to kick back and relax. Not so quickly! Our job is far from complete once we've deployed. We need to continue to invest resources in the listed areas. Each is discussed in detail in the following slides.

DW Lifecycle- Step 5: Maintenance and Growth

Support

- Critical to hook the user.
- For first several weeks, work with users.
- No news is NOT good news.
- Relocate to the users if needed.
- If problems uncovered, be honest, immediate action to fix.
- If deliverable is of not high quality, rework could be substantial.

User support
It is crucial immediately following the deployment in order to ensure that the business community gets hooked. For the first several weeks following user education, the support team should be working proactively with the users. Relocate (at least temporarily) to the business community so that the users have easy access to support resources. If problems with the data or applications are uncovered, be honest with the business to build credibility while taking immediate action to correct the problems. If your warehouse deliverable is not of high quality, the unanticipated support demands for data reconciliation and application rework can be overwhelming.

DW Lifecycle - Step 5: Maintenance and Growth

Education

- Continuing education program.
- Formal refresher, as well as advanced courses and repeat introductory course.
- Informal education for developers and power users for exchange of ideas.

Education
We need to provide a continuing education program for the data warehouse. The curriculum should include formal refresher and advanced courses, as well as repeat introductory courses. More informal education can be offered to the developers and power users to encourage the interchange of ideas.

DW Lifecycle - Step 5: Maintenance and Growth

Technical support

- No longer nice-to-have, but to be treated as a production environment.
- Performance to be monitored proactively and usage trends noted.
- Business users not responsible to tell that the performance has degraded.

Technical support
The data warehouse is no longer a nice-to-have but needs to be treated as a production environment, complete with service level agreements. Of course, technical support should proactively monitor performance and system capacity trends. Don't want to rely on the business community to tell that performance has degraded.

DW Lifecycle - Step 5: Maintenance and Growth

Program support

- A DWH phase may wind-down, but the DWH program lives.
- Market your success.
- Ensure implementation on track and address to business needs.
- Ongoing checkpoint to be implemented.
- Don't loose focus, else failure.

Program support
We need to continue monitoring progress against the agreed-on success criteria. We need to market our success. We also need to ensure that the existing implementations remain on track and continue to address the needs of the business. Ongoing checkpoint reviews are a key tool to assess and identify opportunities for improvement with prior deliverables. Data warehouses most often fall off track when they lose their focus on serving the information needs of the business users.

Lecture No. 35
DWH Life Cycle: Pitfalls, Mistakes, Tips

In this lecture we will discusses the problems, troubles and mistakes that commonly occur while building a data warehouse. We will discuss things to do, and also things not to do. We will discuss ways to avoid common mistakes that may halt data ware housing process.

5 Signs of trouble

1. Project proceeded for two months and nobody has touched the data.
2. End users are not involved hands-on from day one throughout the program.
3. IT team members doing data design (modelers and DBAs) have never used the access tools.
4. Summary tables defined before raw atomic data is acquired and base tables have been built.
5. Data design finished before participants have experimented with tools and live data.

Signs of trouble

First of all we will discuss "5 signs of trouble". Any of these signs if present, will serve as a key indicator that the data warehousing project is under threat. The following situations indicate a project in trouble:

- The project has proceeded for two months and nobody has even touched the data. Before even embarking on the project, the team should have had a thorough understanding and look and feel of the data. As I always tell my students, "know your data intimately".

- The future consumers are not involved hands-on from day one throughout the program. Working in isolation will result in systems that no one will accept, and consequently none is going to use, resulting in negative marketing for you and your company. Thus avoid this at all costs.

- The team members doing data design (modelers and DBAs) have never used the access tools. You need experienced campaigners not "green apples" or raw hands. You need people who know the job, not those who will learn on the job, and in the process sink the ship.

- The summary tables are defined before the raw atomic data is acquired and base tables have been built. A converse process has been followed, which again is a recipe for disaster.

- The data design is finished before participants have experimented with the tools and live data. As we have discussed at length in lecture no. 33, involve the business users from the very beginning, get user requirement definition, record it and follow it.

11-Possible Pitfalls

- **1. Weak business sponsor:** Getting stuck by office politics, need CXO on your side.

- **2. Not having multiple servers**: Penny wise pound Foolish. (i) Both going down (ii) Performance degradation

- **3. Modeling without domain expert:**

- **4. Not enough time for ETL:** Giving too little time or over complicating by including everything conceivable. Users will forgive:
    - Less Formatting, slow system, few features, few reports BUT NOT incorrect results

11-Possible pitfalls in DWH Life Cycle & Development

Many early data warehousing projects failed, having fallen into one or more of the traps we will discuss. These pitfalls are still difficult to avoid, unless those steering the project are able to understand and anticipate the associated risks.

1. Weak business sponsor
This phase often turns out to be the trickiest phase of the data warehousing implementation and is also the Part-II(a) of your semester project. Because data warehousing by definition includes data from multiple sources spanning many different departments within the enterprise. Therefore, there are often political battles that center on the willingness of information sharing. Even though a successful data warehouse benefits the enterprise, there are occasions where departments may not feel the same way. As a result of unwillingness of certain groups to release data or to participate in the data warehousing requirements definition, the data warehouse effort either never gets off the ground, or could not get started in the right direction defined originally. When this happens, it would be ideal to have a strong business sponsor. If the sponsor is at the CXO level (X: Information, Knowledge, Financial etc), he/she can often exert enough influence to make sure everyone cooperates.

2. Not having multiple servers
This is a classical example of penny wise and pound foolish. To save capital, often data warehousing teams will decide to use only a single database and a single server for the different environments i.e. development and production. Environment separation is achieved by either a directory structure or setting up distinct instances of the database. This is awkward for the following reasons:

- Sometimes it is possible that the server needs to be rebooted for the development environment. Having a separate development environment will prevent the production environment from being effected by this.

- There may be interference while having different database environments on a single server. For example, having multiple long queries running on the development server could affect the performance on the production server, as both are same.

3. Modeling without domain expert
It is essential to have a subject-matter expert as part of the data modeling team. This person can be an outside consultant or can be someone in-house with extensive industry experience. Without this person, it becomes difficult to get a definitive answer on many of the questions, and the entire project gets dragged out, as the end users may not always be available.

4. Not enough time for ETL
This is common everywhere, ETL getting the least time, remember data is always dirtier than you think. There is a tendency to give this particular phase of DWH too little time and other

resources. This can prove suicidal to the project, as the end users will usually tolerate less formatting, longer time to run reports, less functionality (slicing and dicing), or fewer delivered reports; one thing that they will never ever tolerate is wrong information.

A second common problem is that some people unnecessarily make the ETL process complicated. In ETL design, the primary goal should be to optimize load speed without sacrificing on quality. This is, however, sometimes not followed. There are cases when the design goal is to cover all possible future uses and possible scenarios, some of which may be practical, while others just plain impractical. When this happens, ETL perfo rmance suffers, and often so does the performance of the entire data warehousing system.

11-Possible Pitfalls (continued)

- **5. Low priority for OLAP Cube Construction**: Giving it the lowest priority.
- **6. Fixation with technology:** End users impressed by timely information NOT advanced infra-structure.
- 7. **Wrong test bench:** Required performance NOT on fast production level machines.
- 8. QA people NOT DWH literate: Ensure QA people are educated about DWH.

5. Low priority for OLAP Cube Construction

Make sure your OLAP cube-building or pre-calculation process is optimized and given the right priority. It is common for the data warehouse to be on the bottom of the nightly batch loads, and after the loading the DWH, usually there isn't much time left for the OLAP cube to be refreshed. As a result, it is worthwhile to experiment with the OLAP cube generation paths to ensure optimal performance.

6. Fixation with technology

Just remember that the end users do not care how complex or how technologically advanced your front end (or for that matter back-end) infrastructure is. All they care is that they should receive their information in a timely manner and in the way they specified.

7. Wrong test bench

Make sure the development environment is very similar to the production environment as much as possible - Performance enhancements seen on less powerful machines sometimes do not happen on the larger, production-level machines.

8. QA people NOT DWH literate

As mentioned above, usually the QA team members know little about data warehousing, and some of them may even resent the need to have to learn another tool or tools. Make sure the QA team members get enough education about data warehousing so that they can complete the testing themselves.

11-Possible Pitfalls (continued)

- **9. Uneducated user:** Take care and address end-user education needs. Intuition does not work.

- **10. Improper documentation:** Complete documentation before developers leave.
- **11. Doing incremental enhancements:** Definite no-no. Dev => QA =>Production

9. Uneducated user
Take care and address the user education needs. There is nothing more frustrating to spend several months to develop and QA the data warehousing system, only to have little usage because the users are not properly trained and educated. Regardless of how intuitive or easy the interface may be, it is always a good idea to send the users to at least a one-day training course to let them understand what they can achieve by properly using the data warehouse.

10. Improper documentation
Usually by this time most, if not all, of the developers will have left the project, so it is essential that proper documentation is left for those who are handling production maintenance. There is nothing more frustrating than staring at something another person did, yet unable to figure it out due to the lack of proper documentation.

Another pitfall is that the maintenance phase is usually boring. So, if there is another phase of the data warehouse planned, start on that as soon as possible.

11. Doing incremental enhancements
Because a lot of times the changes are simple to make, it is very tempting to just go ahead and make the change in production. This is a definite no -no. Many unexpected problems will pop up if this is done. It is very strongly recommend that the typical cycle of **Development** ® **QA** ® **Production** be followed, regardless of how simple the change may seem.

Pitfall: Searching for the "Silver Bullet"

Abandon completely even the desire to find a silver bullet.

- Wasting time on that one access tool that will handle all needs of all users.
- Many tools in the market, instead do a match-making.
- Beware: Vendors use ambiguous, confusing, non-standard nomenclature, which sometimes serves their own purpose.
- For any meaningful comparison, evaluate tools by classifying on the basis of functionality.

Pitfall: Searching for the "Silver Bullet": Pitfalls of Selecting the Tools
Abandon completely even the desire to find a silver bullet.

Many data warehouse project teams waste enormous amounts of time searching in vain for a silver bullet i.e. a panacea or *Amratdhara*. They believe their mission is to find the one access tool that will handle all the needs of all their users. Don't even try. It can't be done. One size does not fit all.

There is a wide and ever-growing diversity of tools for data access, exploration, analysis, and presentation. The appropriate mission of a data warehouse or decision support team is to understand these diverse alternatives and properly match the tool to the intended usage.

The names typically applied to tools are ambiguous and confusing. Generally every new name only obscures the issue more. What is the definitive definition of ad hoc query tool, decision support system, executive information system or online analytic processing? Some terms, like EIS, carry the burden of years of overzealous selling and underwhelming results.

To evaluate tools, you need to slot the alternative into categories that allow for meaningful comparison. Since the traditional terms add little discriminatory power, where do you turn? The first part of the answer is to create purely functional categories.

Pitfall: Extremes of Tech. Arch. Design

Common mistake: Attacking the problem from two extremes, neither is correct.

- Focusing on data warehouse delivery, architecture feels like a distraction and impediment to progress and often end up rebuilding.
- Investing years in architecture, forgetting primary purpose is to solve business problems, not to address any plausible (and not so plausible) technical challenge.

Pitfall: Extremes of Tech. Arch. Design Data warehouse teams approach the technical architecture design process from opposite ends of the spectrum. Some teams are so focused on data warehouse delivery that the architectures feels like a distraction and impediment to progress and eventually, these teams often end up rebuilding. At the other extreme, some teams want to invest two years designing the architecture while forgetting that the primary purpose of a data warehouse is to solve business problems, not address any plausible (and not so plausible) technical challenge. Neither end of the architecture spectrum is healthy; the most appropriate response lies somewhere in the middle.

Top 10-Common Mistakes to Avoid…

- **Mistake 1**: *Not interacting directly with the end users.*
- **Mistake 2**: *Promising an ambitious data mart as the first deliverable.*
- **Mistake 3**: *Never freezing the requirements i.e. being an accommodating person.*
- **Mistake 4:** *Working without senior executives in loop, waiting to include them after a significant success.*
- **Mistake 5:** *Doing a very comprehensive and detailed first analysis to do the DWH right the very first time.*

10-Common Data warehouse mistakes to avoid

So far you have been told what to do, however now we'll balance those recommendations with a list of what not to do. When building and managing a data warehouse, the common mistakes to avoid are listed. These mistakes are described as a series of negative caricatures. The goal is for you to learn from these as George Santayana said, "Those who cannot remember the past are condemned to repeat it." Let's all agree not to repeat any of these mistakes. Each of the mistakes will be discussed one by one.

Mistake 1: *Not interacting directly with the end users;* your job is to be the publisher of the right data. To achieve your job objectives, you must listen to the business users, who are always right. Nothing substitutes for direct interaction with the users. Develop the ability to listen.

Mistake 2: *Promising an ambitious data mart as the first deliverable;* these kinds of data marts are 'consolidated, second-level marts with serious dependencies on multiple sources of data. Customer profitability requires all the sources of revenue and all the sources of cost, as well as an allocation scheme to map costs onto the revenue! For the first deliverable, focus instead on a single source of data, and do the more ambitious data marts later.

Mistake 3: *Never freezing the requirements i.e. being an accommodating person;* You need to think like a software developer and manage three very visible stages of developing each data mart: (1) the business requirements gathering stage, where every suggestion is considered seriously, (2) the implementation stage, where changes can be accommodated~ but must be negotiated and generally will cause the schedule to slip, and (3) the rollout stage, where project features are frozen. In the second and third stages, you must avoid insidious scope creep (and stop being such an accommodating person).

Mistake 4: *Working without senior executives in loop, waiting to include them after a significant success;* the senior executives must support the data warehouse effort from the very beginning. If they don't, your organization likely will not be able to use the data warehouse effectively. Get their support prior to launching the project.

Mistake 5: *Doing a very comprehensive and detailed first analysis to do the DWH right the very first time;* Very few organizations and human beings can develop the perfect comprehensive plan for a data warehouse upfront. Not only are the data assets of an organization too vast and complex to describe completely, but also the urgent business drivers will change significantly over the life of the data warehouse. Start with lightweight data warehouse bus architecture of conformed dimensions and conformed facts, and then build your data warehouse iteratively. You will keep altering and building it forever.

Top 10-Common Mistakes to Avoid

- **Mistake 6:** *Assuming the business users will develop their own "killer application" on their own.*
- **Mistake 7:** *Training users on the detailed features of the tool using dummy data and consider it a success.*
- **Mistake 8:** *Isolating the IT support people from the end or business users.*
- **Mistake 9:** *After DWH is finished, holding a planning and communications meeting with end users.*
- **Mistake 10:** *Shying away from operational source systems people, assuming they are too*

busy.

Mistake 6: *Assuming the business users will develop their own "killer application" on their own;* Business users are not application developers. They will embrace the data warehouse only if a set of prebuilt analytic applications is beckoning them.

Mistake 7: *Training users on the detailed features of the tool using dummy data and consider it a success;* Delay training until your first data mart is ready to go live on real data. Keep the first training session short, and focus only on the simple uses of the access tool. Allocate more time to the data content and analytic applications rather than to the tool. Plan on a permanent series of beginning training classes and follow-up training classes as well. Take credit for the user acceptance milestone when your users are still using the data warehouse six months after they have been trained.

Mistake 8: *Isolating the IT support people from the end or business users;* Data warehouse support people should be physically located in the business departments, and while on assignment, they should spend all their waking hours devoted to the business content of the departments they serve. Such a relationship engenders trust and credibility with the business users.

Mistake 9: *After DWH is finished, holding a planning and communications meeting with end users;* Newsletters, training sessions, and ongoing personal support of the business community should be to gather items for the first rollout of the data warehouse.

Mistake 10: *Shying away from operational source systems people, assuming they are too busy;* Certainly, they cannot alter their operational procedures significantly for passing data to or from the warehouse. If your organization really understands and values the data warehouse, then the operational source systems should be effective partners with you in downloading the data needed and in uploading cleaned data as appropriate.

Top 7-Key Steps for a smooth DWH implementation…

- **Step-1:** Assigning a full-time project manager, or doing it yourself full-time.
- **Step-2:** Consider handing-off project management.
- **Step-3:** During user interview don't go after answers, let the answers come to you.

7-Tips to a smooth data warehouse implementation

There's a long list of things not to do in managing a data warehouse project, but there are also a number of positive, proactive steps that can increase your chances of a smooth implementation. Resolve to be open to new ideas, and seek creative inspiration in radically modifying your tried-and-true practices to fit this new way of thinking.

1. Assigning a full-time project manager, or doing it yourself full-time

It's common, and often unavoidable, that project managers ride herd on several projects at once. The economics of IT resources make this a fact of life. When it comes to building a data warehouse, however, don't even think about it. You are entering a domain unlike anything else you and your crew have worked on. Everything about it—analysis, design, programming, testing, modifications, maintenance—will be new. You, or whoever you assign as project manager, will have a much better shot at success if allowed to get into that "new" mode and stay there.

2. Consider handing-off project management
Because the phases of a data-warehouse build are so very different, you do yourself no disservice by handing off to another project manager when a phase is complete, provided you adhere to Step One above. Why is it reasonable to do this? First, any phase of a data warehouse implementation can be exhausting, from a project management standpoint. From the deployment of physical storage to implementing the Extract-Transform-Load, from designing and developing schemas to OLAP, the phases of a warehouse build are also markedly different from one another. Each not only could use a fresh hand, management-wise, but a fresh creative perspective. Handing off management not only doesn't necessarily hurt, it may even help.

3. During user interviews don't go after answers let the answers come to you.
This is important enough to be an article in itself. You must understand, going into the design process that your potential warehouse users aren't going to be able to clearly articulate what it is they want the warehouse to do for them. They're going to have to explore and discover it as they go—and so will your development team, in conducting interviews. Make your interviews open-ended, with lots of note-taking, and have your development-team interviewer's focus more on the consequences of processes than the processes themselves.

Since you're conducting these interviews in order to get some idea of what data to store and how to efficiently store it, you need to (in partnership with your users) come up with new ways to look at data, not process it. You're trying to find potential information that can be gleaned, not from transactional data itself, but from the information behind it: the rise and fall of numbers over time, etc. Don't chase answers in these interviews. Let answers come to you.

Top 7-Key Steps for a smooth DWH implementation

- **Step-4:** Assigning responsibilities to oversee and ensure continuity.
- **Step-5:** Accept the "fact" that DWH will require many iterations before it is ready.
- **Step-6:** Assign significant resources for ETL.
- **Step-7:** Be a diplomat NOT a technologist.

4. Assigning responsibilities to oversee and ensure continuity.
These don't need to be full-time assignments, but because the phases of a data warehouse implementation differ so greatly, you're going to need people out there assuring continuity. There are three important areas: (i) architecture, (ii) technology, and (iii) business. Assign an architecture lead to ensure that the generally agreed-upon architecture of the data warehouse, from the physical level on up, is maintained throughout the project. A technology lead should be appointed, because your developers and key users will all be using tools they've never used before—someone needs to oversee the deployment and consistent use of these tools.

Finally, the business needs that will be met through use of the warehouse must be carefully observed and documented, to spur continued development. Since the analytics and metrics to be derived from the process are developed over time, by users who will not necessarily communicate well with one another, someone must watch this development, encourage its continuation, and nurture it into progressing to higher levels.

5. Accept the "fact" that DWH will require many iterations before it is ready.
A data warehouse will never, ever be right the first time. Why? You don't know what you're really looking for until you see it. Or, to say it more precisely, the ultimate users of the system won't know what they're really going to use it for until they've used it for awhile. As contrary as that may seem to all that you've sworn by throughout your career, it really is the way to go: business intelligence is an infant science, and different for every company.

You'll have to fish around for the right data in the right format, and things will change often. BI is very "personal," unique to your environment, your market, and your partnerships. What does this mean? First of all, it means you need to lock your database administrator in a room somewhere and break the news that the data warehouse data structures are going to change and change and change, as will the ETL procedures. There is no way around this. Make your peace with it now, and save both yourself and the DBA a lot of stress.

6. Assign significant resources for ETL
You're going to be stepping in it again and again as you wade through oceans of old data, in old databases, on old magnetic tape, from remote sources. Much of it will be dirty. Much of it will be hard to get to. You're going to be doing a lot of this, and you're going to be devising ETL procedures to seek out and retrieve information like this forevermore. You do yourself and the project a great service by establishing a method of doing this right the first time. Have your development people put in the extra time to explore old data thoroughly, characterize "dirty" data issues realistically, and to design and implement robust extraction and transformation procedures exhaustively. The ETL portion of a data warehouse can consume as much as 80 percent of your total project resources! Make sure you spend wisely.

7. Be a diplomat NOT a technologist
The biggest problem you will face during a warehouse implementation will be people, not the technology or the development. You're going to have senior management complaining about completion dates and unclear objectives. You're going to have development people protesting that everything takes too long and why can't they do it the old way? You're going to have users with outrageously unrealistic expectations, who are used to systems that require mouse-clicking but not much intellectual investment on their part. And you're going to grow exhausted, separating out Needs from Wants at all levels. Commit from the outset to work very hard at communicating the realities, encouraging investment, and cultivating the development of new skills in your team and your users (and even your bosses).

Most of all, keep smiling. When all is said and done, you'll have a resource in place that will do magic, and your grief will be long past. Eventually, your smile will be effortless and real.

Conclusions

- DWH is not simple.
- DWH is very expensive.
- DWH is not ONLY about technology.

- DWH designers must be capable of working across the organization.
- DWH team requires a combination of many experiences and expertise.

Conclusions

By now you have realized that a building a data warehouse is not an easy task. DWH are very expensive to build, with the average cost of a system valued at around US$ 2 million. Hence, the right people, methodology and experience is critical. The dependence on technology is only a small part in realizing the true business value buried within the mountain of data collected and stored within organizations business systems and operational databases. Data warehouses touch the organization at all levels, and the people that design and build the data warehouse must be capable of working across the organization at all levels as well, thus communication skills of the people are of utmost importance. Thus the key requirements are industry and product experience of a diverse team, coupled with a business focus and proven methodology. This will make the difference between just a functional system and true success story.

Lecture-36
Course Project

<u>This will be a group project, consisting of not more than three students per group.</u>

Two Parts

- Part-I outline given in 1st lecture. (about coding)
- Part-II briefly mentioned in 33rd lecture. (about planning and req definition)

Objective: Give a head-start by identifying the work in advance.

Approach: Combine both parts.

The course project consists of two parts, and both parts have already been briefly touched upon in the earlier lectures. The first part of the project was discussed in the very first lecture and we call it Part-I, while the second part of the project was discussed in lecture no. 33 and we will call it Part-II. In this lecture, we will discuss both parts in detail. The purpose of an early discussion was to give you a head start on the projects so that you start thinking/working about them, such as polishing your programming skills for Part-I and identifying the organization for Part-II. Our approach will be to combine both parts of the projects to come up with a single semester project.

Part-I

1. Code the BSN Method for finding siblings and duplicates (Lect-20).
2. Use 4GL or a high level programming language.
3. Must have GUI for I/P and O/P
4. For input, use data of Lahore campus provided as part of lab work.
5. Submit project report_1.

Part-I of the project basically deals with the implementation of the BSN (Basic Sorted Neighborhood) Method that we discussed in lecture no. 20. You already have the paper that was discussed in the said lecture. As you know that the BSN method can be used for dedupication i.e. removal of duplicates, this is one aspect that you will have to code and test. The other application of BSN will be to identify the siblings, people who have the same father i.e. they are brothers or sisters of each other. To implement this part of the project, you can either use a 4GL such as PL/SQL or any other high level programming language such as C++, or Java. Your application must have a GUI (Graphical User Interface), console based application will not be acceptable. The user must have the facility of reading from a data source, which could be a text file or a database or a spreadsheet. In case of a database or spread sheet, OLE DB connectivity must be supported by your application.

There can be basically two options for the data input. Either you can use data collected as part of Project-II (after approval of instructor) or use the text files supplied for the Lahore campus as part of the SQL DTS (Data Transformation Services) Lab Lectures. If you use data other than the lab data, it must have a dozen relevant columns and several thousand rows too.

Part-II: Implementation Life-Cycle

The second part of the semester project i.e. Part-II deals with the DWH implementation life cycle that we have already discussed in great detail in lectures 32-35. In lectures 33 and 34 we discussed the Ralph Kimball's approach, you are not required to do any development and deployment work in project Part-II. During the lectures, the DWH life cycle road-map was divided into three parts, you only have to cover these parts i.e. (i) project planning (ii) user requirement definition and (iii) three parallel tracks. You are NOT required to discuss or do DWH deployment or do any analytics development.

Part-II(a): Identify Organization

1. Do a complete data warehouse implementation life cycle study (and design) as discussed in lectures 32-35.
2. Identify a large company/organization that is a prime candidate for a DWH.
3. What is a large company/organization? Lot of customers and large number of transactions.
4. Submit report_2 giving and explaining 4-reasons for selecting a company.
5. Get the company/organization selected approved by the instructor before proceeding ahead.

As stated earlier, the lifecycle study for project Part-II ends after the three parallel tracks of the Kimball's approach; this was also stated in lecture 33. You are not required to do any deployment or the study of deployment. Part-II of the project consist of two parts i.e. Part-II(a) which consists of identification of a company for the lifecycle study, and Part-II(b) actually doing the lifecycle study. To do Part-II(a), you have to identify a large company or an organization, this could be private or government owned, but must be a prime candidate for such a study.

A shop at the corner of your street with ship-owner being the salesman too is not a prime target for a DWH. He may actually do not even need a database system. So what is a large company or an organization? Well one which has large number of customers and large number of transactions. So what is large is this circular logic or a trick question? Neither. Large customers mean tens of thousands of customers and similarly tens and thousands of transactions per week. Note that what we call large on our country, may be small for the developed world. Once you have identified such a company, submit a report, we call this report_2. The report should have four reasons why you have selected a particular company or organization. What could be those four reasons?, this is a good question, the four reasons are (i) the number of customers (ii) the number of transactions (iii) typical early adopter (from next slide) and (iv) any other reason. Once you have submitted the report, you can not just by default move on to Part-II(b) of the project. The company selected must be approved by the instructor before you can proceed ahead.

Large and Typical Early Adopters

1. Financial service/insurance.
2. Telecommunications.
3. Transportation.
4. Government.
5. Educational.

Here you would be asking the question, what is meant by an early adopter? Will discuss this at the end of the lecture, but if you can't wait, please check fig-36.1. For a developing country like ours, we are in the phase of talking about typical early adopters of DWH. However, in the developed world; this stage is no longer there for many large companies, as DWH are now in the mainstream. The types of organizations and business listed are typically those that have a large number of customers and large number of transactions.

Example DWH Target Organizations

- Financial service/insurance.
  - Union Bank
  - State Bank of Pakistan
- Telecommunications.
  - UFone
  - PTCL
  - PAKNET
- Transportation.
  - PIA
- Government.
  - NADRA

For example, as per *www.paksearch.com* Muslim Commercial Bank has 900+ branches all over Pakistan. With an average of 500 customers per branch, the total number of customers is in the order of half a million. It would not be surprising if the weekly ATM transactions all over Pakistan run into millions. Such banks are potential candidates for a data warehouse. Same is true for telecommunication companies. As per recent government figures, there are 10+ million mobile phone users in Pakistan, and as per *www.fdi.com* the number of mobile phone users of Mobilink is 3.7 million. Again, it would not be surprising to have literally millions of mobile phone calls made/received per day. So these businesses fall under the category which you should be looking at to select and study as part of your semester project.

NADRA (National Database and Registration Authority) probably has the largest data warehouse in Pakistan and is the repository of the 1998 census which covered the entire population of Pakistan, which at time stood at 130 millions. As part of the census (source: *www.nadra.gov.pk*) 64 million NDFs or National Data Forms were collected and scanned which are presently stored in a 4.2 Tera byte DWH in NADRA.

Part-II(b): Life-Cycle Study…

Project Planning

- By now you already know the company.
- Prepare a questionnaire (at least 15 non-trivial questions).
- Identify and contact a key person who will help you.

Submit report_3

User requirement definition

Set an appointment to meet business users.

- Collect answers to questions.
- Get a copy of sample input.
- Get a copy of sample output.
- Compile report of interview.
- Identify business processes.
- Identify problems.
- Identify measures of success.

Submit report_4

Once report_2 has been approved, you are all set to move on to Part-II(b) of the project i.e. the actual lifecycle case study. The first part of this study is project planning, that we discussed in great detail in lecture no. 33. Do the necessary preparations, which include development of a questionnaire of atleast 15 questions before you meet the key person in the company. The questions MUST be non-trivial i.e. must not be overly simplistic i.e. asking for information already available in the company brochure or their website. Before preparing the questionnaire, you must have done the necessary background study about the company, so that you make relevant and probing questions. You must also have identified and set an appointment with the key person who is willing to help you. Note that any person will not suffice, but a person who is in a position of influence and wants to help you. IF you can't find such a person, don't worry, keep on reading, I will provide you guidance on how to resolve this problem. After you have successfully gone through the project planning phase, you should submit report_3.

Now the next part is user requirement definition, we have discussed this too in great detail in lecture no. 33. I would suggest that before you meet the user for requirement collection, call and set an appointment, reach there in time and in formal attire. Also take along a micro cassette reorder or use your mobile phone for recording the session. Do the recording with permission from the business user. Along with recording the answers of the user (don't forget to take notes) get a copy of the sample input used in the organization, and a copy of the typical output, that could be in the form of a report etc. Specifically ask questions and understand the business process, the problems and the measures of success. After you are done, go through the entire process of debriefing etc, and submit report_4.

Part-II(b): Life-Cycle Study…

Do a technical track study and submit report_5.

Do a data track design and submit report_6.

Do an analytic track study and submit report_7.

About reports:

- Each report A4 page size and NOT more than 5 pages (excluding diagrams and tables).
- 12-pt Times New Roman font
- Single space and 1" top, bottom, left and right spacing

What will be covered in report 5 through 7 was covered in detail in lecture no. 34, read the notes and the reference book for these lectures i.e. "The Data Warehouse Toolkit by Ralph Kimball". The only difference in the project work w.r.t to the lecture is that you don't have to do the analytics development, and don't have to go beyond the three parallel tracks.

What if you are <u>not</u> entertained?

- There are prospects, then there are customers.
- Typically 1 out of 10 prospects is a buyer or customer i.e. 10% success.
- The more prospects you meet and they are not interested, the more close you actually get to your customer.

    - Quitters never win, and winners never quit.

- Thomas Edison made 2,000 attempts to make the filament for the light bulb that actually worked.
- When asked how did he managed 2,000 failures? He said they were not failures; he just identified WHICH 2,000 filaments don't work.
- DON'T GIVE UP

In the context of sales and marketing, a prospect is someone who can become a customer. A customer is someone who will buy a product or service form you. So every sales person is looking for a customer or wants to convert a prospect into a customer. However, on an average 10% of the prospects become a customer, it means if you try 10 prospects, in the worst case the last one will be the customer. Look at positively, meaning the more failures you have, the more close you get to your customer. This is why the saying goes that "quitters never win, and winners never quit". Similarly look at the work of Thomas Edison, he kept trying and untimely become immortal i.e. his name still lives and his invention still used.

What if you are never entertained?

It may so happen that no one entertains/helps you, maybe you did not tried hard, maybe you had the wrong attitude, maybe you did not meet the right people, maybe you were not very convincing, maybe the end user was apprehensive etc.

Upto 10% less credit if this approach is adopted
In such a case, search the web, read books/magazines and pick any one of the 5 types of organizations discussed and collect reference/related material (not beyond year 2000).

Use the material collected to write reports 2 to 6.

Now it may so happen, that you tried, and tried and tried very hard, but still unable to get a key person interested and willing to help you so that you could write to write report_2. There could be several reasons for this, maybe you don't tried hard, maybe it was because of the other person etc. In such a case, you will do a lifecycle case study using the Internet. In report_2 you will list all the reasons for failure and how you tried, and then we will decide about how much credit to deduct from the semester project. The amount of deduction could be upto 10%. But you still have to write all the reports and use the material from the case studies tailored to the requirements of the reports. For this purpose you will have to download a number of focused case studies, better if about the same organization, and compile the results in the form of the reports as per the Ralph Kimball's road map.

Contents of Project Reports

The project reports to include, but is not limited to, the following:

- Narrative summary of results produced (report_1).
- Listings of computer models/programs coded and utilized (report_1).
- Reports displaying results (report_1).
- System usage instructions (report_1).
- Narrative description of business and tables of appropriate data (report_4).
- Descriptions of decisions to be supported by information produced by system (report_4).

- Structure charts, dataflow diagrams and/or other diagrams to document the structure of the system (report 4-7).
- Recommendations (reports 5-7)

This is self explanatory, and I have also explained in the lecture. Follow the guidelines to the word, as your work will be graded based on these guidelines. Note that system usage instructions have to be specific with screen shots of the application developed, so that your applications can be executed using the instructions and graded. Don't forget to submit the entire source code, along with the compiled code with all necessary libraries DLLs attached.

Format of Project Reports: Main

- Report No.
- Title of course, semester & submission date
- Names and roll no. of project members.
- Campus and name of city.
- Table of contents.
- 1-page executive summary for each report.
- Attach (scanned) hard/soft copies of all related material collected and referenced with each report.
- At the end of semester, combine ALL reports and submit as a single report.

Again this is self explanatory, and I have also explained in the lecture. Follow the guidelines to the word, as your work will be graded based on these guidelines.

Format of Project Reports: Other

- No spelling or grammar mistakes.
- Make sections and number them (as per contents of report discussed).
- Pages should be numbered (bottom center).
- Add an index at the end of report.
- File name: RPT_no_semester_campus_rollno_CS614.doc
- e.g. RPT_1_F05_BestComputersWah_234,235_CS614.doc
- Email copy of report.
- **Warning:**
  - Do not copy-paste, I can and I will catch you.

MS Word has a facility to create an index. You begin by creating a file with a list of keywords to be listed in the index, and then go to Insert then Reference then Index and Tables, press AutoMark and select the file with keywords.

The naming convention of the reports is important, so that your reports can be easily identified as there are about half a dozen reports for each group. The following convention to be used in report naming:

RPT: This will be repeated for each report and will not change.

no: is the report number i.e. 1 through 7

semester: is the semester, some possible semesters are F05 i.e. Fall 2005 or SP06 i.e. Spring 2006 or SPL05 i.e. special semester 2005 or SM06 i.e. summer 2006.

campus: The name of the VU campus where you are registered or taking the course along with the name of the city or town. Write full name of the campus, do not use underscore i.e. _ or dashes or space in the campus name.

rollno: Since it is a group project, so roll numbers of students in the group separated by a comma.

CS614: This will be at the end of every file name i.e. the course code.

Don’t try to copy-paste or use someone else’s work with your name, there are smart tools to catch this, once you are caught, this can result into zero credit.

Why would companies entertain you?

- You are students, and whom you meet were also once students.
- You can do an assessment of the company for DWH potential at no cost.
- Since you are only interested in your project, so your analysis will be neutral.
- Your report can form a basis for a professional detailed assessment at a later stage.
- If a DWH already exists, you can do an independent audit.

If you present your case well you are likely to be entertained i.e. welcome by the organization. The first and the foremost reason to get help is, whom you are talking to was once a student too as you are currently, so there is a common bond. Since you have studied well DWH (hopefully), therefore, you can do a requirement assessment (in the form of lifecycle study) of the company at no cost; the company has nothing to lose. Your only interest is completion of your project and grade, so you are going to be very objective and very neutra l; hence the company has still nothing to lose. After you have done the lifecycle study, the same can be used as a seed or input by a professional organization for an in depth study, thus saving in money to the company for which you have done the work. Again the company has nothing to lose. It may so happen that the company you contact already has a data warehouse in place, in such a case doing the lifecycle development study can be used as a internal audit of the DWH implementation. Hence in short, if the company allows and helps you with the study, it will be a win-win scenario for both the parties.

Why you may be entertained?

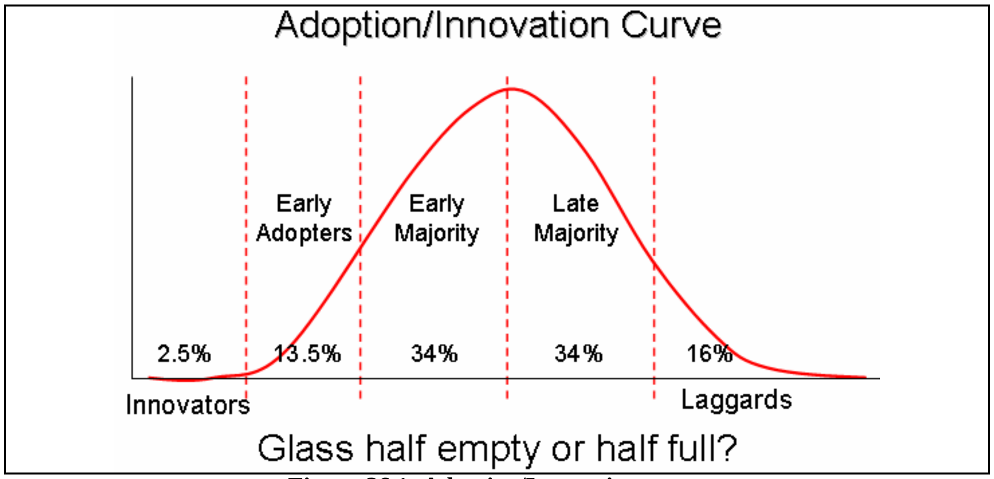


Figure-36.1: Adoption/Innovation curve

Fig-36.1 shows a typical adoption curve when a new item, product or service is introduced and the ratio of people responding to it. As you can see the people who are the first ones to adopt or embrace it are the innovators, and they are very few. This is followed by early adopters, which is a sizeable figure, and this is the category of companies you are supposed to target as part of your project. Note that in our country, Data Warehousing is not yet in the category of early majority, so there will be more companies who are prime candidates for lifecycle study. You may also come across people who may be genuinely interested in a DWH, but don't know enough about it, and want to know. In such a case, be prepared to educate them or enlighten them. It would be best if you look at a number of case studies or reports about their line of business before meeting them.

Interestingly, the bell shaped curve (which is not a perfect bell) divides the prospects into two equal parts i.e. 50% each. You should be looking at that 50% which can and will help you, instead of those who are likely to help you, but at a later stage.

Finally this project has enough challenges to become your final year BSc project. In such a case be prepared to do coding leading to system deployment and completion of all the remaining steps. Remember, many large organizations/businesses may not need a data warehouse today, but they will need one surely tomorrow. And when that happens, they will be looking for you to help them achieve their objectives.

<u>The project has more than enough potential to become a final year project, which will cover an implementation and deployment also.</u>

Step 6: Data Acquisition & Cleansing

Trained scouts from DPWQCP periodically visit randomly selected points and manually note 35 attributes, with some given in Table 2. These hand-written sheets are subsequently filed. For the last 10 years, the data collected was recorded by typing the hand-filled pest scouting sheets. Copy of a hand filled pest scouting sheet is shown in Figure -38.1(a).

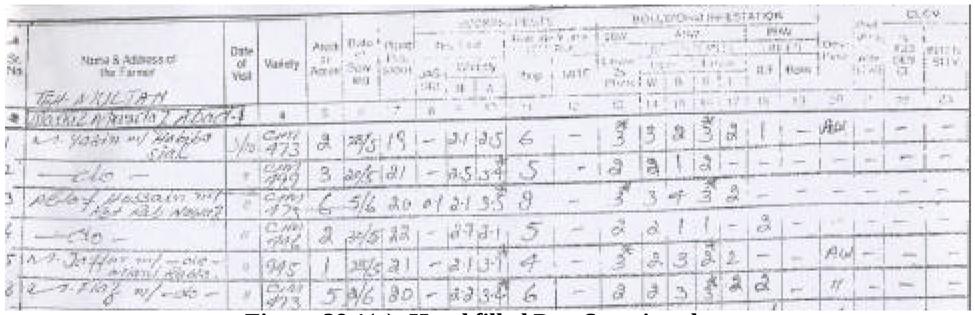
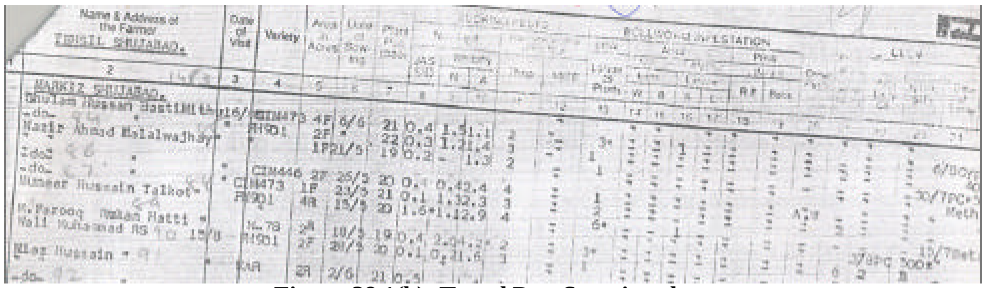

Sr. No. | Name & Address of the Farmer | Date of Visit | Variety | Area in Acres | Date of Sowing | Plant Pop. | BOLLWORM INFESTATION

Figure-38.1(a): Hand filled Pest Scouting sheet


Name & Address of the Farmer | Date of Visit | Variety | Area in Acres | Date of Sowing | BOLLWORM INFESTATION

TEHSIL SHUJABAD.

MARKIZ SHUJABAD.

Ghulam Hussan Dastimith
-do-
Nazir Ahmad Malalwajhay
-do-
-do-
Muneer Hussain Talkot
M.Farooq
Wali Muhammad
Niaz Hussain
-do-

Figure-38.1(b): Typed Pest Scouting sheet

The * in Figure-38.1 corresponds to pest hot spot or flare-up or ETL_A.

Step-6: Issues

- The pest scouting sheets are larger than A4 size (8.5” x 11”), hence the right end was cropped when scanned on a flat-bed A4 size scanner.
- The right part of the scouting sheet is also the most troublesome, because of pesticide names for a single record typed on multiple lines i.e. for multiple farmers.
- As a first step, OCR (Optical Character Reader) based image to text transformation of the pest scouting sheets was attempted. But it did not work even for relatively clean sheets with very high scanning resolutions.
- Subsequently DEO’s (Data Entry Operators) were employed to digitize the scouting sheets by typing.

The pest scouting sheets are larger than A4 size (8.5” x 11”), hence the right end was cropped when scanned on a flat-bed A4 size scanner. The right part of the scouting sheet is also the most troublesome, because of pesticide names for a single record typed on multiple lines i.e. for multiple farmers.

As a first step, OCR (Optical Character Reader) based image to text transformation of the pest scouting sheets was attempted. But it did not work even for relatively clean sheets with very high scanning resolutions, such as 600 dpi. Subsequently DEO’s (Data Entry Operators) were employed to digitize the scouting sheets by typing. To reduce spelling errors in pesticide names and addresses, drop down menu or combo boxes with standard and correct names were created and used.

Step-6: Why the issues?

- Major issues of data cleansing had arisen due to data processing and handling at four levels by different groups of people
    1. Hand recordings by the scouts at the field level.
    2. Typing hand recordings into data sheets at the DPWQCP office.
    3. Photocopying of the typed sheets by DPWQCP personnel.
    4. Data entry or digitization by hired data entry operators.

Data cleansing and standardization is probably the largest part in an ETL exercise. For Agri-DWH major issues of data cleansing had arisen due to data processing and handling at four levels by different groups of people i.e. (i) Hand recordings by the scouts at the field level (ii) typing hand recordings into data sheets at the DPWQCP office (iii) photocopying of the scouting sheets by DPWQCP personnel and finally (iv) data entry or digitization by hired data entry operators.

After achieving acceptable level of data quality, the data was loaded into Teradata data warehouse; subsequently each column was probed using SQL for erroneous entries. Some of the errors found were correct data in wrong columns, nonstandard or invalid variety names etc. There were some intrinsic errors, such as variety type “999” or spray_date "12:00:00 AM" inserted by the system against missing values. Variations found in pesticide names and cotton variety names were removed by comparing them with standard names.

Step 7: Data Transform, Transport & Populate

Among the different types of transformations performed in the implementation, only the more complex i.e. multiple M:1 transformations for field individualization will be discussed in this section.

Motivation for Transformation

- Trivial queries give wrong results.
- Static and dynamic attributes
- Static attributes recorded repeatedly.

Table 2 gives details of the main attributes recorded at each point. Static attributes are those attributes that are recorded on each visit by the scouts, usually does not changes.

Static Attributes		Dynamic Attributes	
1	Farmer Name	1	Date of Visit
2	Farmer Address	2	Pest Population
3	Field Acreage	3	CLCV
4	Variety(ies) Sown	4	Predator Population
5	Sowing date	5	Pesticide Spray Dates
6	Sowing method	6	Pesticide(s) Used

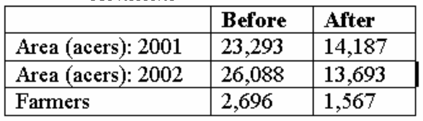
Table-38.1: Cotton pest scouting attributes recorded by DPWQCP surveyors

The data recorded consists of two parts i.e. static and dynamic (Table -38.1). On each visit, the static, as well as the dynamic data is recorded by the scouts, thus resulting in static values getting recorded repeatedly. Since no mechanism is used to uniquely identify each and every farmer, therefore, trivial queries, such as total area scouted, distribution of varieties sown etc. gives wrong results. For example, while aggregating area, the area of the farmer with multiple visits during the season is counted multiple times, giving incorrect results, same is true for varieties sown. Therefore, to do any reasonable analysis after data cleansing, the most important step of data transformation being individualization of the cultivated fields, not farmers. The reason being, a farmer usually has multiple fields, but a field is associated or owned by a single farmer.

Step-7: Resolving the issue

- **Solution:** Individualization of cultivated fields.
    - Technique similar to BSN used to fix names.
    - Unique ID assigned to farmers.
    - BSN used again, and unique ID assigned to fields.

- **Results:**



	Before	After
Area (acers): 2001	23,293	14,187
Area (acers): 2002	26,088	13,693
Farmers	2,696	1,567

- **Limitation:** Field individualization not perfect. Some cases of farmers with same geography, sowing date, same variety and same area. Such cases were dropped.

Method
Field individualization turned out to be a very laborious process. It was attempted by first uniquely identifying the farmers. This was achieved by collectivity sorting farmer name, *Mozua* and *Markaz*. The grouping of farmer names was scrutinized to fix the spelling errors in the farmer names and unique farmer_ID was assigned to each farmer. Subsequently based on the farmer_ID, sowing date, area and variety, cultivated fields were uniquely identified and field_ID assigned to each field.

Results
To demonstrate the amount of error removed because of field individualization, consider the case of scouted area and unique farmers. Without field individualization, the cotton scouted area for 2001 and 2002 added to **23,293** and **26,088** acres, respectively. After field individualization, the correct scouted area turned out to be **14,187** and **13,693** acres respectively i.e. a correction of about 50%. Similarly unique farmers reduced from **2,696** to **1,567**. The method of field

individualization is in no way perfect, there were some cases of farmers with same geography, sowing date, same variety and same area. Such cases were dropped.

Transporting the data
Once the data entry was complete, double checked and reconciled the corresponding files were compressed and moved from the premises of the DEO (Data Entry Operator) to the University, where sample printout of data entered were taken and a final random quality check was performed. Subsequently minor errors, if any were fixed and data was loaded into the Agri-DWH.

Step 8: Determine Middleware Connectivity
Since the source data is maintained in a non digital format, hence connectivity with the data warehouse was irrelevant. Once digitized, it was rather trivial to load the data into the warehouse. Furthermore, in the foreseeable future, it was not anticipated that the scouting sheets were going to be maintained in a digitized form.

Steps 9-11: Prototyping, Querying & Reporting

- Implemented the prototype with user involvement.

- Applications developed
    - **10.** A data mining tool was also developed based on an indigenous technique that used crossing minimization paradigm for unsupervised clustering.

    - **11.** A low-cost OLAP tool was indigenously developed; actually it was a Multi dimensional OLAP or MOLAP.

    - Use querying & reporting tools
    - The following SQL query was used for validation:

```
SELECT Date_of_Visit, AVG(Predators),
                                        AVG(Dose1+Dose2+Dose3+Dose4)
FROM Scouting_Data
WHERE Date_of_Visit < #12/31/2001# and predators > 0
GROUP BY Date_of_Visit;
```

Implement a prototype with user involvement
The Agri-DWH was implemented with the involvement of the end users. In this regard there was close collaboration between the development team and personnel of (i) Directorate of Pest Warning, Multan (ii) National Agriculture Research Center (NARC), Islamabad (iii) Pakistan Agriculture Research Council (PARC) and (iv) Agriculture University, Faisalabad. The implementation was centered around numerous meetings with the potential end users, discussion of results, and also explicit set of questions provided by them.

Applications developed
A low-cost OLAP tool was indigenously developed; actually it was a Multi dimensional OLAP or MOLAP. Using the MOLAP tool, agriculture extension data was analyzed. A data mining tool was also developed based on an indigenous technique that used crossing minimization paradigm for unsupervised clustering.

Use querying & reporting tools

Despite small number of rows i.e. 4,400, the Agri-DWH was implemented using Teradata for the sake of completion of the entire cycle. The following SQL query was used to generate Figure-38.2.

```
SELECT Date_of_Visit, AVG(Predators), AVG(Dose1+Dose2+Dose3+Dose4)
FROM Scouting_Data
WHERE Date_of_Visit < #12/31/2001# and predators > 0
GROUP BY Date_of_Visit;
```

Step 12: Deployment & System Management
Since Agri-DWH was a pilot project, therefore, the traditional deployment methodologies and system management techniques were not followed to the word, and are not discussed here.

Decision Support using Agri-DWH

Agri-DSS usage: Data Validation

- Quality and validity of the underlying data is the key to meaningful and authentic analysis.
- After ensuring a satisfactory level of data quality (based on cost-benefit trade-off) extremely important to scientifically validate the data that the DWH will constitute.
- Some very natural checks were employed for this purpose. Relationship between the pesticide spraying and predator (beneficial insects) population is a fact well understood by agriculturists.
- Predator population decreases as pesticide spray increases and then continually decreases till the end of season.

Quality and validity of the underlying data is the key to meaningful and authentic analysis. After ensuring a satisfactory level of data quality (based on cost-benefit trade-off) it is extremely important to somehow judge the validity of data that a data warehouse constitutes. Some very natural checks were employed for this purpose. Relationship between the pesticide spraying and predator (beneficial insects) population is a fact well understood by agriculturists. Predator population decreases as pesticide spray increases and then continually decreases till the end of season, as shown in Fig-38.2. In Figure-38.2 the y-axis shows the relative frequency of pesticide sprays in multiple of 100 ml, and average predators population greater than zero.

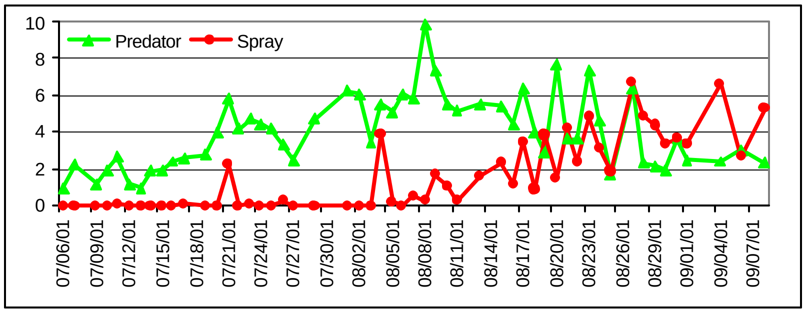


Figure 38.2: Year 2001 Frequency of spray Vs. Predators population

FAO Report

Pesticides are used as means for increasing production, as a positive correlation is believed to exist between yield and pesticide usage. However, existence of an undesirable, sometime even negative correlation between pesticide usage and yield has been observed in Pakistan (FAO: Food and Agriculture Organization report 2001). Figure-38.3 shows a marked decrease in yield while the pesticide usage is on the rise, and also its converse, creating a complex situation.

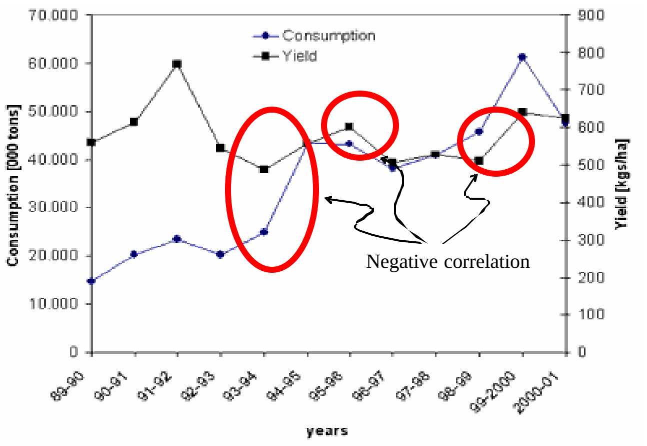


Figure 38.3: Yield and Pesticide Usage in Pakistan: Source FAO (2001)

Excessive use of pesticides is harmful in multiple ways. On one hand, farmers have to pay more for the pesticides, while on the other, increased pesticide usage develops immunity in pests, thus making them more harmful to the crops. Excessive usage of many pesticides is also harmful for the environment and hazardous to human.

Reasons for pesticide abuse can be discovered by automatically exploring pest scouting and metrological data.

Working Behaviors at Field Level: Spray dates

As expected, the results of querying for spray dates and spray frequency for 2001 and 2002 do not display any well defined patterns; as it is dependent on pest populations (Fig -37.2), availability of pesticides etc. To study the relationship between sprays and time, moving average of sprays for five days, and a moving correlation of sprays for five days were calculated. For the sake of uniformity, the moving average of spray was normalized using the maximum spray frequency. The results are shown in Figure -38.4.

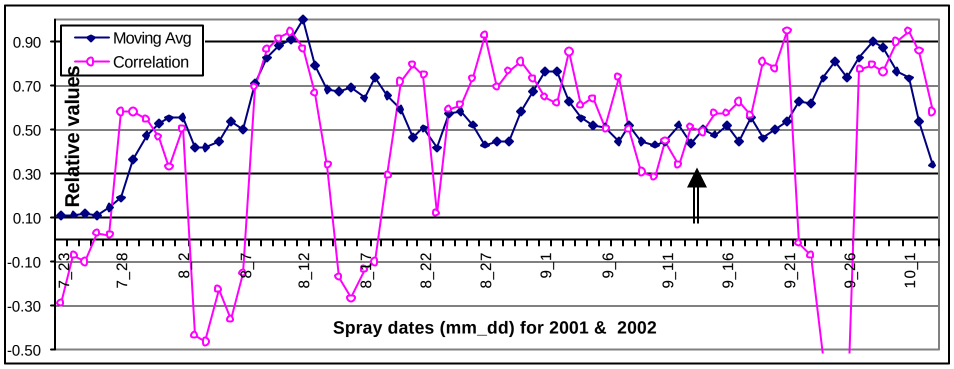


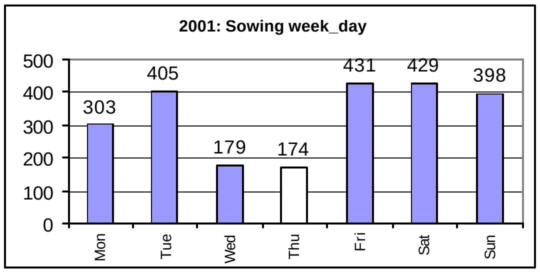
Figure-38.4(a): Spray frequency Vs. day of year for Year 2001

No relationship should have existed for the two years. But note the surprising finding that most sprays occurring on and around 12th Aug. in BOTH years with high correlation, appearing as a spike. Also note the dip in sprays around 11th Sep.! Sowing at predetermined time makes sense, as it is under the control of the farmer, but that is not true for spraying. Pests don't follow calendars; therefore, whenever, ETL_A is crossed pesticides are sprayed.

14th Aug. is the independence day of Pakistan and a national holiday. In Pakistan, people are in a habit of sandwiching gazetted holidays with casual leaves; consequently businesses are closed for a longer period, including that of pesticide suppliers. 14th Aug. occurred on Tue and Wed in 2001 and 2002, respectively, thus making it ideal to stretch the weekend. During Aug/Sep. humidity is also high, with correspondingly high chances of pest infestations. Therefore, apparently the farmers decided not to take any chances, and started spraying around 11th Aug.; evidently even when it was not required. Unfortunately, the weather forecast for 13 Aug. 2001 and 2002 was showers and cloudy, respectively. Therefore, most likely the pesticide sprayed was washed-off. Unfortunately the decline in sprays around 9/11 could not be explained.

Working Behaviors at Field Level: Sowing dates

The results of querying the sowing date based on the day of the week are shown in Fig-38.5.



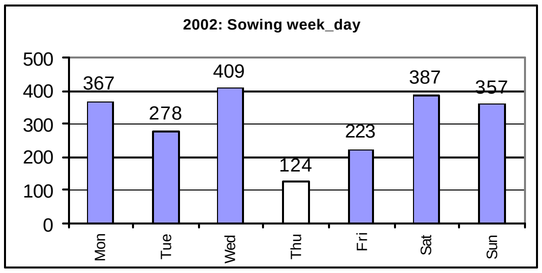


Figure 38.5: Number of sowings against week days

Observe least number of sowings done on Thursdays, in each year. This finding was later confirmed by extension personnel. Multan is famous for its shrines. Thursdays are usually related with religious festivals and activities, a mix of devotion and recreation, and usually held at shrines, hence a tendency of doing less work on Thursdays. Similar behavior was observed for spraying too.

Conclusions & lessons learnt

- Extract Transform Load (ETL) of agricultural extension data is a big issue. There are no digitized operational databases so one has to resort to data available in typed (or hand written) pest scouting sheets. Data entry of these sheets is very expensive, slow and prone to errors.

- Particular to the pest scouting data, each farmer is repeatedly visited by agriculture extension people. This results in repetition of information, about land, sowing date, variety etc (Table-2). Hence, farmer and land individualization are critical, so that repetition may not impair aggregate queries. Such an individualization task is hard to implement for multiple reasons.

- There is a skewness in the scouting data. Public extension personnel (scouts) are more likely to visit educated or progressive farmers, as it makes their job of data collection easy. Furthermore, large land owners and influential farmers are also more frequently visited by the scouts. Thus the data does not give a true statistical picture of the farmer demographics.

- Unlike traditional data warehouse where the end users are decision makers, here the decision-making goes all the way “down” to the extension level. This presents a challenge to the analytical operations’ designer, as the findings must be fairly simple to understand and communicate.

Lecture-39
Web Warehousing: An introduction

Through the billions of Web pages created with HTML and XML, or generated dynamically by underlying Web database service engines, the Web captures almost all aspects of human endeavor and provides a fertile ground for data mining. However, searching, comprehending, and using the semi structured information stored on the Web poses a significant challenge because this data is more sophisticated and dynamic than the information that commercial database systems store. To supplement keyword-based indexing, which forms the cornerstone for Web search engines, researchers have applied data mining to Web-page ranking. In this context, data mining helps Web search engines find high-quality Web pages and enhances Web clickstream analysis. For the Web to reach its full potential, believe this technology will play an increasingly important role in meeting the challenges of developing the intelligent Web.

Putting the pieces together

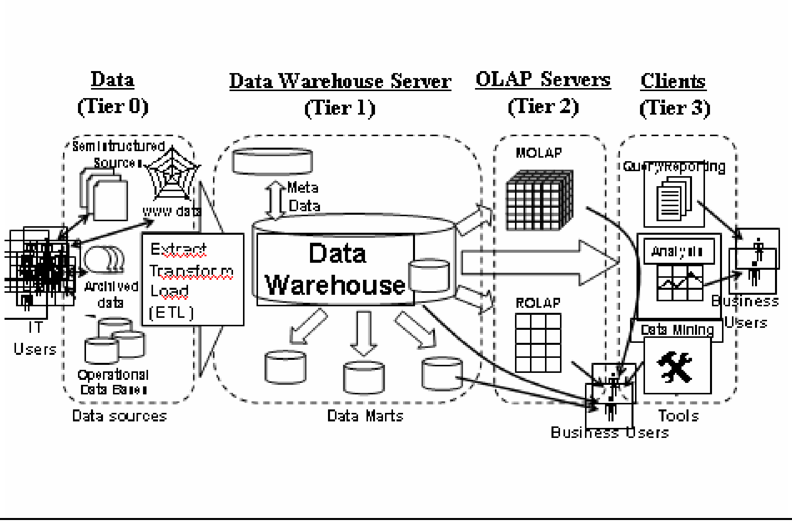


Figure-39.1: Putting pieces together

Figure 39.1 shows the same multi-tier architecture for data warehousing, as was shown in a previous lecture. The figure shows different components of an overall decision support system and their interaction.

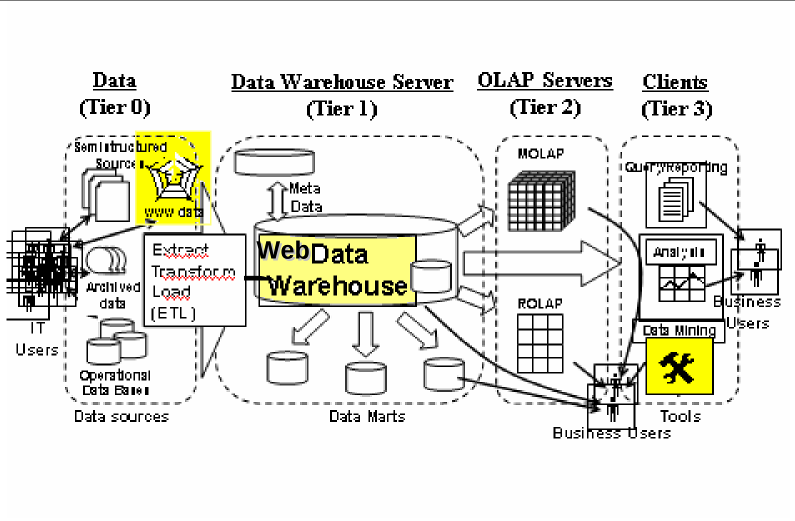


Figure-39.2: Putting pieces together–Web warehousing

In case of Web warehousing, Figure 39.1 has been slightly modified to highlight components the major components needed, as shown in Figure 39.2. Here the main source i.e. data source is the World Wide Web. The central repository Data Warehouse has changed to a Web data warehouse and data mining tools are customized to cater for the semi structured and dynamic web data.

Background

Internet population stands at around a billion users.

Exponential growth rate of web and size. Indexable web is more than 11.5 billion pages (Jan 2005).

Adoption rate of intranet & extranet warehouse having similar growth rate

Web enabled versions of tools are available, but adoption differ.

Media is div erse and other than only data i.e. text, image, audio etc.

We are all familiar with the growth rate of the public Web. Regardless of the metric used to measure its growth attached networks, servers, users or pages the growth rate continues to exhibit an exponential pattern. In the same vein, the adoption rate of intranet and extranet data warehouses (i.e., Web warehouses) has exhibited a similar pattern, although the pattern has lagged public adoption. While data warehouse and business intelligence vendors have offered Web-enabled versions of their tools for the past few years, and the sales of their Web offerings began to substantially exceed the sales of their traditional client/server versions. However, the patterns of adoption differ by more than a simple lag.

The media delivered by a Web warehouse include not only data, but text, graphics, image, sound, video, and other forms as well.

Reasons for web warehousing

1. Searching the web (web mining).
2. Analyzing web traffic.
3. Archiving the web.

The three reasons for warehousing web data are as listed in the slide. First, web warehousing can be used to mine the huge web content for searching information of interest. Its like searching the golden needle from the haystack. Second reason of Web warehousing is to analyze the huge web traffic. This can be of interest to Web Site owners, for e-commerce, for e-advertisement and so on. Last but not least reason of Web warehousing is to archive the huge web content because of its dynamic nature. As we proceed we will discuss all theses concepts in further detail.

Web searching

Web is large, actually very large.

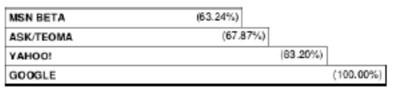
To make it useful must be able to find the page(s) of interest/relevance.

How can the search be successful?

Three major types of searches, as follows:

1. Keyword-based search
2. Querying deep Web sources
3. Random surfing

The success of google



The Web—an immense and dynamic collection of pages that includes countless hyperlinks and huge volumes of access and usage information—provides a rich and unprecedented data mining source. How can a search identify that portion of the Web that is truly relevant to one user's interests? How can a search find high-quality Web pages on a specified topic?

Currently, users can choose from three major approaches when accessing information stored on the Web:

(i) Keyword-based search or topic-directory browsing with search engines such as Google or Yahoo, which use keyword indices or manually built directories to find documents with specified keywords or topics;

(ii) Querying deep Web sources—where information, such as amazon.com's book data and realtor.com's real-estate data, hides behind searchable database query forms—that, unlike the surface Web, cannot be accessed through static URL links; and

(iii)Random surfing that follows Web linkage pointers.

The success of these techniques, especially with the more recent page ranking in Google and other search engines shows the Web's great promise to become the ultimate information system.

Drawbacks of traditional web sear ches

1. Limited to keyword based matching.
2. Can not distinguish between the contexts in which a link is used.
3. Coupling of files has to be done manually.

Data warehousing concepts are being applied over the Web today. Traditionally, simple search engines have been used to retrieve information from the Web. These serve the basic purpose of data recovery, but have several drawbacks. Most of these engines are based on keyword searches limited to string matching only. That narrows down our retrieval options. Als o we have links, at times several levels of them in a particular context. But simple search engines do not do much justice to obtaining information present in these links. They provide direct information recovery, but not enough indirect link information. Also if we have files related to certain subjects and need to couple these, the coupling has to be done manually. Web search engines do not provide mechanisms to incorporate the above features. These and other reasons have led to further research in the area of Web knowledge discovery and have opened the window to the world of Web Warehousing..

Why web warehousing-Reason no. 1?

Web data is unstructured and dynamic, keyword search is insufficient.

To increase usage of web, must make it more comprehensible.

Data Mining is required for understanding the web.

Data mining used to rank and find high quality pages, thus making most of search time.

The Web with billions of Web pages provides a fertile ground for data mining. However, searching, comprehend ing, and using the semi structured information stored on the Web poses a significant challenge because this data is more sophisticated and dynamic than the information that commercial database systems store.

To supplement keyword -based indexing, which forms the cornerstone for Web search engines, researchers have applied data mining to Web-page ranking. In this context, data mining helps Web search engines find high-quality Web pages and enhances Web clickstream analysis. For the Web to reach its full potential, however, we must improve its services, make it more comprehensible, and increase its usability. As researchers continue to develop data mining techniques, we believe this technology will play an increasingly important role in meeting the challenges of developing the intelligent Web.

Why web warehousing-Reason no. 2?

Web log contains wealth of information, as it is a key touch point.

Every customer interaction is recorded.

Success of email or other marketing campaign can be measured by integrating with other operational systems.

Common measurements are:
- Number of visitors
- Number of sessions
- Most requested pages
- Robot activity
- Etc.

The Web log contains a wealth of information about the company Web site, a touch point for e-business. The Web log, or clickstream, gathers this information by recording every interaction a customer or potential customer has with the business over the Internet. The success of a specific e-mail, marketing or ad campaign can be directly measured and quantified by integrating the Web log with other operational systems such as sales force automation (SFA), customer relationship management (CRM) and enterprise resource planning (ERP) applications.

Common measurements include number of visitors, number of sessions, most requested page, most requested download, most accessed directories, leading referrer sites, leading browser and operating system, visits by geographic breakdown, and many others. This information can be used to modify the design of the Web site, change ad campaigns or develop partnering relationships with other sites. While these metrics are insightful and important for managing the corporate e-business on a short term or tactical basis, the real value of this knowledge comes through integration of this e-resource with other customer touch-point information.

Why web warehousing-Reason no. 3?

Shift from distribution platform to a general communication platform.

New uses from e-government to e-commerce and new forms of art etc. between different levels of society.

Thus web is worthy to be archived to be used for several other projects.

Such as snapshots of preserving time.

In recent years, we have seen not only an incredible growth of the amount of information available on the Web, but also a shift of the Web from a platform for distributing information among IT-related persons to a general platform for communication and data exchange at all levels of society. The Web is being used as a source of information and entertainment; forms the basis for e-government and e-commerce; has inspired new forms of art; and serves as a general platform for meeting and communicating with others via various discussion forums. It attracts and involves a broad range of groups in our society, from school children to professionals of various disciplines to seniors, all forming their own unique communities on the Web. This situation gave rise to the recognition of the Web's worthiness of being archived, and the subsequent creation of numerous projects aiming at the creation of World Wide Web archives. Snapshot-like copies of the Web preserve an impression of what hyperspace looked like at a given point in time, what kind of information, issues, and problems people from all kinds of cultural and sociological backgrounds were interested in, the means they used to communicate their interests over the Web, characteristic styles of how Web sites were designed to attract visitors, and many other facets of this medium.

How Client Server interactions take place?

- Understanding the interactions essential for understanding the source and meaning of the data in the clickstream
- Sequence of actions for the browser and Web site interaction using HTTP

Understanding the interactions between a Web client (browser) and a Web server (W eb site) is essential for understanding the source and meaning of the data in the clickstream. In Figure in next slide, we show a browser, designated "Visitor Browser." We'll look at what happens in a typical interaction from the perspective of a browser user. The browser and Web site interact with each other across the Internet using the Web's communication protocol-the Hypertext Transfer Protocol (HTTP).

How Client Server interactions take place?
Illustration

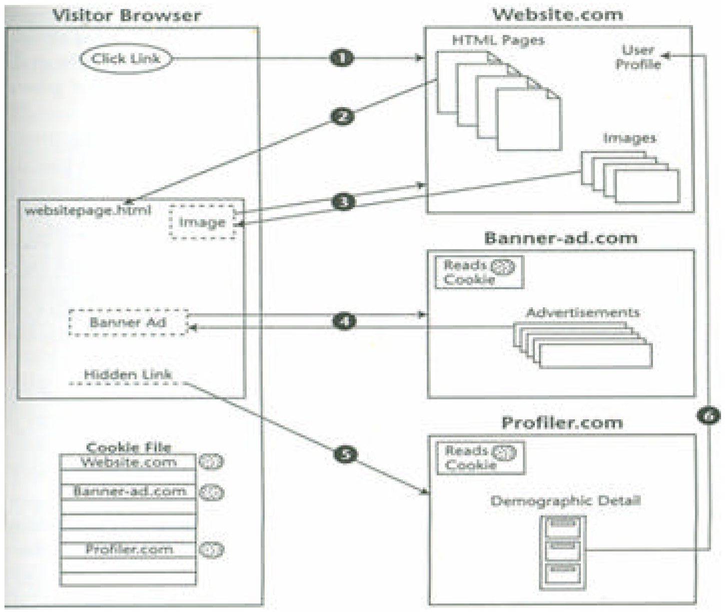


Figure-39.3: Illustration of how Client Server interactions

Figure 39.3 illustrates the steps during a client server interaction on the WWW. Each of the activity or action has been shown with a sequence . Lets briefly look at these sequence of actions.

Action 1
User tries to access the site using its URL.

Action 2
The server returns the requested page, websitepage.html. Once the document is entirely retrieved, the visitor's browser scans for references to other Web documents that it must fulfill before its work is completed. In order to speed up the response time, most browsers will execute these consequential actions in parallel, typically with up to 4 or more HTTP requests being serviced concurrently.

Action 3
The visitor's browser finds a reference to a logo image that is located at Website. Com. The browser issues a second request to the server, and the server responds by returning the specified image.

Action 4
The browser continues to the next reference for another image from Banner-ad.com. The browser makes this request, and the server at Banner-ad.com interprets a request for the image in a special way. Rather than immediately sending back an image, the banner-ad server first issues a cookie request to the visitor's browser requesting the contents of any cookie that might have been placed previously in the visitor's PC by Banner-ad.com. There are two options based on the response of the cookie request;
Option I: *Cookie Request Fulfilled:* The banner-ad Web site retrieves this cookie and uses the contents as a key to determine which banner ad the visitor should receive. This decision is based on the visitor's interests or on previous ads. Once the banner-ad server makes a determination of the optimal ad, it returns the selected image to the visitor. The banner-ad server then logs which ad it has placed along with the date and the clickstream data from the visitor's request.
Option II**:** *No Cookie Found: If the* banner-ad server had not found its own cookie, it would have sent a new persistent cookie to the visitor's browser for future reference, sent a random banner ad, and started a history in its database of interactions with the visitor's browser.
Referrer: The *HTTP* request from the visitor's browser to the banner-ad server carried with it a key piece of information known as the *referrer.* The referrer is the URL of the agent responsible for placing the link on the page. In the example the referrer is Website.com/websitepage.html. Because Banner-ad.com now knows who the referrer was, it can credit Website. com for having placed an advertisement on a browser window.

Action 5
In the original HTML document, websitepage.html had a hidden field that contained a request to retrieve a specific document from Profiler.com. When this request reached the profiler server, Profiler.com immediately tried to find its cookie in the visitor's browser. This cookie would contain a user ID placed previously by the profiler that is used to identify the visitor and serves as a key to personal information contained in the profiler's database.

Action 6
The profiler might either return its profile data to the visitor's browser to be sent back to the initial Web site or send a real-time notification to the referrer, Website.com, via an alternative path alerting Website.com that the visitor is currently logged onto Website. com and viewing a specific page. This information also could be returned to the HTML document to be returned to the referrer as part of a query string the next time an HTTP request is sent to Website.com.

Although Figure 39.3 shows three different sites involved in serving the contents of one document, it is possible, indeed likely, that these functions will be combined into fewer servers. It is likely that advertising and profiling will be done within the same enterp rise, so a single request (and cookie) would suffice to retrieve personal information that would more precisely target the ads that are returned. However, it is equally possible that a Web page could contain references to different ad/profile services, providing revenue to the referrer from multiple sources.

Low level web traffic Analysis

Is anyone coming?

If people are coming, identify which pages they are viewing.

Once you rank your content, you can tailor it to satisfy your visitors.

Detailed analysis helps increase traffic, such which sites are referring visitors.

There are many reasons why you might want to analyze your Web site's traffic. At the lowest level, you want to determine if anyone is coming to your Web site in order to justify the site's existence.

Once you have determined that people are indeed visiting your Web site, the next step is to identify which pages they are viewing. After determining what content is popular and what content is ignored, you can tailor your content to satisfy your visitors.

High level web traffic analysis

Combining your Web site traffic data with other data sources to find which banner ad campaigns generated the most revenue vs. just the most visitors.

Help calculate ROI on specific online marketing campaigns.

Help get detailed information about your online customers and prospects.

A more detailed analysis of your Web site traffic will assist you in increasing the traffic to your Web site. For example, by determining which sites are referring visitors to your site, you can determine which online marketing activities are most successful at driving traffic to your site. Or, by combining your Web site traffic data with other data sources, such as your customer databases, you can determine which banner ad campaigns, for example, generated the most revenue versus just the most visitors. This allows you to calculate the return on investment (ROI) on specific online marketing campaigns as well as get detailed information about your online customers and prospects.

What information can be extracted?

Identify the person by the originating URL if filled a form.

Came through search engine, or referral, if search engine using which keyword.

Viewing which pages, using which path and how long a view.

Which visitors spent the most money…

Thus a lot to discover.

First, you can determine who is visiting your Web site. Minimally, you can determine what company the person is from (the host computer that they are using to surf the Web—ford.com would be the Ford Motor Company, for example). Additionally, if a visitor filled out an online form during a visit to your Web site, you can link the form data with his or her Web site traffic data and identify each visitor by name, address, and phone number (and any other data that your online forms gather).

You can also learn where your visitors are coming from. For example, did they find your site by using a search engine such as Google or did they click on a link at another site? If they did use a search engine, which keywords did they use to locate your site?

Furthermore, you can identify which pages your Web site visitors are viewing, what paths they are taking within your site, and how long they are spending on each page and on the site. You can also determine when they are visiting your site and how often they return.

At the highest level, you can determine which of your Web site visitors spent the most money purchasing your products and services and what the most common paths and referring pages were for these visitors.

As you can see, you can discover a great deal about your Web site visitors—and we only touched upon a few introductory topics.

Where does traffic info. come from?

1. Log files.
2. Cookies.
3. Network traffic.
4. Page tagging.
5. ISP (Internet Service Provider)
6. Others

<u>To track traffic on a web site</u>

http://www.alexa.com/data/details/traffic_details?q=&url=http://www.domain.com

The principal sources of web traffic are as follows:

1. Log files.
2. Cookies.
3. Network traffic.

4. Page tagging.
5. ISP

Others
We will not discuss all of them.

ISPs
Besides log files, we may get clickstream data from referring partners or from Internet service providers (ISPs). If we are an ISP providing Web access to directly connected customers, we have a unique perspective because we see every click of our familiar captive visitors that may allow much more powerful and invasive analysis of the end visitor's sessions
Others
We also may get clickstream data from Web-watcher services that we have hired to place a special control on certain Web pages that alert them to a visitor opening the page. Another important form of clickstream data is the search specification given to a search engine that then directs the visitor to the Web site.

Web traffic record: Log files

Connecting to a web site c onnects to web sever that serves files.

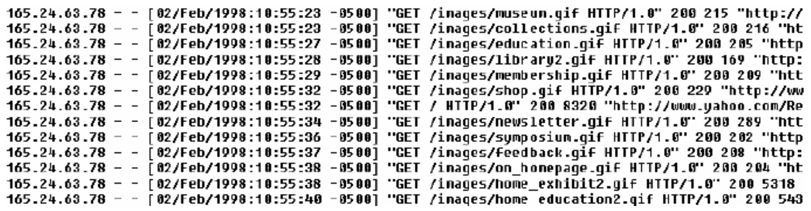
Web server records each action in a text file.

In raw log file is useless, as unstructured and very large.

Many formats and types of log files, can make your own too.

Analyzers can be configured to read most log files.

Web traffic record: Log file format



```
165.24.63.78 - - [02/Feb/1998:10:55:23 -0500] "GET /images/museum.gif HTTP/1.0" 200 215 "http://
165.24.63.78 - - [02/Feb/1998:10:55:23 -0500] "GET /images/collections.gif HTTP/1.0" 200 216 "ht
165.24.63.78 - - [02/Feb/1998:10:55:27 -0500] "GET /images/education.gif HTTP/1.0" 200 205 "http
165.24.63.78 - - [02/Feb/1998:10:55:28 -0500] "GET /images/library2.gif HTTP/1.0" 200 169 "http:
165.24.63.78 - - [02/Feb/1998:10:55:29 -0500] "GET /images/membership.gif HTTP/1.0" 200 209 "htt
165.24.63.78 - - [02/Feb/1998:10:55:32 -0500] "GET /images/shop.gif HTTP/1.0" 200 229 "http://ww
165.24.63.78 - - [02/Feb/1998:10:55:32 -0500] "GET / HTTP/1.0" 200 8320 "http://www.yahoo.com/Re
165.24.63.78 - - [02/Feb/1998:10:55:34 -0500] "GET /images/newsletter.gif HTTP/1.0" 200 289 "htt
165.24.63.78 - - [02/Feb/1998:10:55:36 -0500] "GET /images/symposium.gif HTTP/1.0" 200 202 "http
165.24.63.78 - - [02/Feb/1998:10:55:37 -0500] "GET /images/feedback.gif HTTP/1.0" 200 208 "http:
165.24.63.78 - - [02/Feb/1998:10:55:38 -0500] "GET /images/on_homepage.gif HTTP/1.0" 200 204 "ht
165.24.63.78 - - [02/Feb/1998:10:55:38 -0500] "GET /images/home_exhibit2.gif HTTP/1.0" 200 5318
165.24.63.78 - - [02/Feb/1998:10:55:40 -0500] "GET /images/home_education2.gif HTTP/1.0" 200 543
```

This is just a portion of the first 12 lines of a 250,000 line log file (the file scrolls to the left for several more pages).

This is a 10 megabyte log file and it represents one day’s worth of traffic to a low volume Web site.

Figure- 39.4: A sample Log file format

Figure 39.4 shows a sample web log file, indeed a very small portion of the actual log file. Here, 12 lines have been shown and the file can be scrolled to left to see more columns. The complexity of dealing with Web log data can well be understood , and mostly Web can be even more complex and huge than this sample file.

Web log file formats

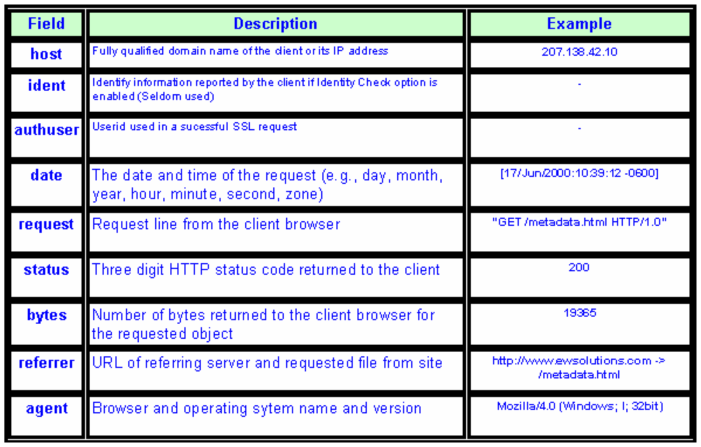
Format of web log dependent on many factors, such as:

- Web server
- Application
- Configuration options

Several servers support CLF ECLF format.

Web log file formats vary depending on the Web server application and configuration options selected during installation. Most Web server applications (including those from Apache, Microsoft and Nets cape) support Common Log file Format (CLF, sometimes pronounced "clog") or Extended Common Log file Format (ECLF). CLF and ECLF formats share the same initial seven fields, but ECLF adds referrer and agent elements.

Web Log File Formats



Field	Description	Example
host	Fully qualified domain name of the client or its IP address	207.138.42.10
ident	Identify information reported by the client if Identity Check option is enabled (Seldom used)	-
authuser	Userid used in a sucessful SSL request	-
date	The date and time of the request (e.g., day, month, year, hour, minute, second, zone)	[17/Jun/2000:10:39:12 -0600]
request	Request line from the client browser	"GET /metadata.html HTTP/1.0"
status	Three digit HTTP status code returned to the client	200
bytes	Number of bytes returned to the client browser for the requested object	19365
referrer	URL of referring server and requested file from site	http://www.ewsolutions.com -> /metadata.html
agent	Browser and operating sytem name and version	Mozilla/4.0 (Windows; I; 32bit)

Table-39.1: Web Log File Formats

Our example proxy log data file contained following fields

i. Timestamp (date in Table 39.1)

ii. Elapsed Time
This is the time that transaction busied the cache. This time is given in milliseconds. For the request where there was a cache-miss this time is minimal, where the request engaged the cache, this time is considerable.

iii. Client Address (host in Table 39.1)

iv. Log Tag
This field tells the result of the cache operation.

v. HTTP Code (status in Table 39.1)

vi. Size (bytes in Table 39.1)

vii. Request Method (request in Table 39.1)
This is the method which client used to initiate the request and be dealt with the proper treatment on the server side.

viii. URL
This is the URL which was request by the client. Therecan be many variations in the representation, start and termination of the URL.

ix. User Ident (ident in Table 39.1)
This field is used to identify the requesting user on the network.

x. Hierarchy Data
This field provides the hierarchy data of the request from the same client in the current request.

xi. Content Type
This field contains the type of data which was requested. The values in this field are the standard MIME types which describe the data contents.

Web traffic record: Cookies

Web log contains one-way traffic record i.e. server to client.

No information about the visitor.

Thus complex heuristics employed that analyze links, time etc. to identify the user.

Cookie is a small text file generated by the web server and stored on the client machine.

This file is read by the web server on a repeat visit.

One of the fundamental flaws of analyzing server log files is that log files contain information about files transferred from the server to the client—not information about people visiting the Web site. For this reason, most log analyzers employ complex heuristics that analyze the raw log data and make educated guesses based on the client's hostname or IP address, the time between page views, the browser type, and the referring page to estimate the number of visits that a site has received. While these estimates are typically good enough, and often better than metrics you can obtain with traditional media, sometimes you simply need more.

A cookie is a small piece of information generated by the Web server and stored on the client machine. Advanced Web site analysis solutions offer the option to use cookies to get much more accurate visitor counts, while also allowing analysis of repeat visitors.

Lecture-40
Web Warehousing: Issues

In the previous lecture we discussed about Web warehousing. The purpose was to get an understanding of the basic concepts of Web warehousing, and its significance. Today we discuss about some of the issues and challenges of the Web warehousing domain. But before staring with issues, lets discuss another reason of why Web warehousing? , in addition to the already discussed three reasons .

Why web warehousing-Reason no. 4?

- **E-Commerce**
  - Data driven economy.
  - Commerce Beyond geographical boundaries.
  - Internet & Credit card required.
  - Many types (B2C, C2C, B2B etc.)
  - Sale/Purchase of non-tangible items.
  - 24x7 operation and service.

Web is different, so is the traffic on it. The traditional equivalent of commercial data warehouse for a web is e-commerce data warehouse. As I said in the very first lecture, we have moved into an information or data driven economy, and commerce is taking place over the internet beyond geographical boundaries.

The world's largest book store **www.amazon.com** is open to business to anyone and everyone with internet access and a credit card. This is typical example of a B2C or business to consumer type of e-commerce. Then there are other types too such as C2C or consumer to consumer e-commerce such as e -bay. Anything and everything is sold, but the quickest items are non-tangible digital items such a music, videos, songs, pictures etc. “Stores” selling such items are open 24x7 and have literally billions of visitors in a month, much more than any traditional OLTP business, hence a prime case for a web warehouse.

Web Warehouse Dimensions

- Some are standard dimensions.
- Other are non-standard and different.
- Some of the dimensions for a Web DWH

Familiar DWH Dim	**W DWH[1] Dim**
Date, Time of day, Part Vendor, Transaction, Status	Page, Event, Session, Referral

Table-40.1: Web Warehouse Dimensions

Some of dimensions for a Web retailer could include:

Date, Time of day, Part, Vendor, Transaction, Status, Service policy, Internal organization, Employee, **Page, Event, Session, Referral** etc.

All the dimensions in the first column of table 40.1 are familiar data warehouse dimensions. The last four in the second column, however, are the unique dimensions of the clickstream and warrant some careful attention. Discussing all of them is beyond the scope of this lecture and this course. Therefore, we will touch upon just one dimension i.e. the page dimension.

Dimensions for W DWH: Page

- Describes the page context for a Web page event
- Definition of page must be flexible
- Assume a well defined function that
  - Characterizes the page
  - Describe the page
- The page dimension is small

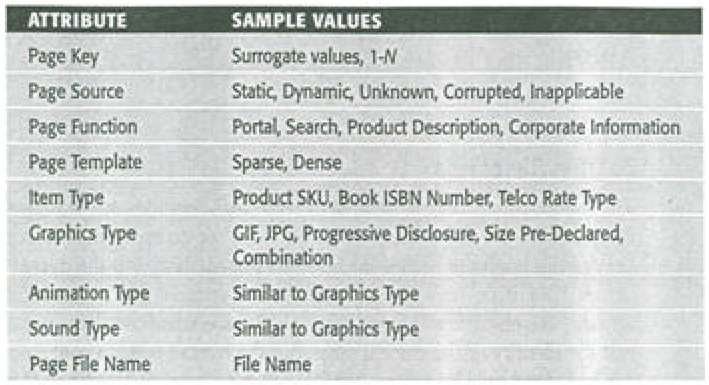
The page dimension describes the page context for a Web page event. The grain of this dimension is the individual page. Our definition of page must be flexible enough to handle the evolution of Web pages from the current, mostly static page delivery to highly dynamic page delivery in which the exact page the customer sees is unique at that instant in time. We will assume even in the case of the dynamic page that there is a well defined function that characterizes the page, and we will use this to describe the page. We will not create a page row for every instance of a dynamic page, because that would yield a dimension with an astronomical number of rows, yet the rows would not differ in interesting ways. What we want is a row in this dimension for each interesting, distinguishable type of page. Static pages probably get their own row, but dynamic pages would be grouped by similar function and type.

When the definition of a static page changes because the Webmaster alters it, the row in the page dimension either can be overwritten or can be treated as a slowly changing dimension. This decision is a matter of policy for the Web data warehouse, and it depends on whether the old and new descriptions of the page differ materially and whether the old definition should be kept for historical analysis purposes.

Web site designers and Web data warehouse developers need to collaborate to assign descriptive codes and attributes to each page served by the Web server, whether the page is dynamic or static. Ideally, Web page developers supply descriptive codes and attributes with each page they create and embed these codes and attributes into the optional fields of the Web log files. This crucial step is at the foundation of the implementation of this page dimension.

The page dimension is small. If the nominal width of a single row is 100 bytes and we have a big Web site with 100,000 pages, then the unindexed data size is 100 x 100,000 = 10 MB. If indexing adds a factor of 3, then the total size of this dimension is about 40 MB.

Details of W DWH Page Dimension



ATTRIBUTE	SAMPLE VALUES
Page Key	Surrogate values, 1-*N*
Page Source	Static, Dynamic, Unknown, Corrupted, Inapplicable
Page Function	Portal, Search, Product Description, Corporate Information
Page Template	Sparse, Dense
Item Type	Product SKU, Book ISBN Number, Telco Rate Type
Graphics Type	GIF, JPG, Progressive Disclosure, Size Pre-Declared, Combination
Animation Type	Similar to Graphics Type
Sound Type	Similar to Graphics Type
Page File Name	File Name

Table-40.2: Details of W DWH Page Dimension

Table40.2 shows some of the common page dimension attributes and their sample values. The number of attributes and also their possible values add to the complexity of modeling Web data.

Clickstream

Clickstream is every page event recorded by each of the company's Web servers

- Web-intensive businesses
- Although most exciting, at the same time it can be the most difficult and most frustrating.

- Not JUST another data source.

Web-intensive businesses have access to a new kind of data, in some cases literally consisting of the gestures of every Web site visitor. This is called as the *clickstream.* In its most elemental form, the clickstream is every page event recorded by the web server. The clickstream contains a number of new dimensions such as page, session, and referrer-that were previously unknown in conventional DWH environment.

The clickstream is a stream of data, easily being the largest text and number data set we have ever considered for a data warehouse. Although the clickstream is the most exciting new development in data warehousing, at the same time it can be the most difficult and most frustrating to handle and process.

The clickstream is not just another data source that is extracted, cleaned, and dumped into the data warehouse. It is an evolving collection of data sources having more than a dozen Web server log file formats for capturing clickstream data. These formats have optional data components that, if used, can be very helpful in identifying visitors, sessions, and the true meaning of behavior.

Issues of Clickstream Data

- Clickstream data has many issues.

  1. Identifying the Visitor Origin
  2. Identifying the Session
  3. Identifying the Visitor
  4. Proxy Servers
  5. Browser Caches

Unlike data from OLTP system, where there were nice user identifications such as unique IDs that were the primary keys, in the context of a web log, this is one of the most issues i.e. identification of the visitor, so is where the visitor actually came from. In OLTP system there was a clean session beginning and session ending, but web is session less. It is very difficult and challenging to identify the session of a visitor, and the list goes on.

Clickstream data contains many ambiguities. Identifying visitor origins, visitor sessions, and visitor identities is something of an interpretive art. Browser caches and proxy servers make these identifications even more challenging.

Identifying the Visitor Origin

- There is no easy way to determine from a log whether or not (your) site is set as a browser's home page of the visitor.

- The site may be reached as a result of a click through----- a *deliberate click on a text or graphic link from another site.*

Lets start with the origin of the visitor.

There is no easy way to determine from a log whether or not your site is set as a browser's home page. This is pretty unlikely unless one is the Webmaster for a portal site or an intranet home page, but many sites have buttons that, when clicked, prompt the visitor to set his or her URL as the browser's home page. Unfortunately, a visitor may be directed to the site from a search at a portal such as Yahoo! or Alta Vista. Such referrals can come either from the portal's index or table of contents, for which placement fee might have been paid, or from a keyword or content search.

The site may be reached as a result of a *click through*---a deliberate click on a text or graphic link from another site. This may be a paid-for referral as via a banner ad or a free referral from an individual or cooperating site. In the case of click-throughs, the referring site almost always will be identifiable in the Web site's referrer log data. Capturing this crucial clickstream data is important to verify the efficacy of marketing programs. It also provides crucial data for auditing invoices one may receive from click-through advertising charges

Identifying the Session

- Web-centric data warehouse applications require every visitor session (visit) to have its own unique identity
- The basic protocol for the World Wide Web, HTTP, stateless so session identity must be established in some other way.
- There are several ways to do this
  - Using Time-contiguous Log Entries
  - Using Transient Cookies
  - Using HTTP's secure sockets layer (SSL)
  - Using session ID Ping-pong
  - Using Persistent Cookies

Most web-centric data warehouse applications will require every visitor session (visit) to have its own unique identity tag similar to a grocery store point-of-sale ticket ID or *session ID*. The rows of every individual visitor action in a session, whether derived from the clickstream or from an application interaction, must contain this tag. However, it must be kept in mind that the operational application generates this session ID, not the Web server.

The basic protocol for the World Wide Web, HTTP, is stateless-that is, it lacks the concept of a session. There are no intrinsic login or logout actions built into the HTTP, so session identity must be established in some other way. There are several ways to do this

1. Using Time-contiguous Log Entries
2. Using Transient Cookies
3. Using HTTP's secure sockets layer (SSL)
4. Using session ID Ping-pong
5. Using Persistent Cookies

1- Using Time-contiguous Log Entries

- A session can be consolidated by collecting time-contiguous log entries from the same host (Internet Protocol, or IP, address).
- Limitations
- The method breaks down for visitors from large ISPs
- Differe nt IP addresses
- Browsers that are behind some firewalls.

In many cases, the individual hits comprising a session can be consolidated by collating time-contiguous log entries from the same host (Internet Protocol, or IP, address). If the log contains a number of entries with the same host ID in a short period of time (for example, one hour), one can reasonably assume that the entries are for the same session.

Limitations

- This method breaks down for visitors from large ISPs because different visitors may reuse dynamically assigned IP addresses over a brief time period.
- Different IP addresses may be used within the same session for the same visitor.
- This approach also presents problems when dealing with browsers that are behind some firewalls.

Notwithstanding these problems, many commercial log analysis products use this method of session tracking, which requires no cookies or special Web server features.

2- Using Transient Cookies

- Let the Web browser place a session-level cookie into the visitor's Web browser.
- Cookie value can serve as a temporary session ID
- **Limitations**
  - You can't tell when the visitor returns to the site at a later time in a new session.

Another, much more satisfactory method is to let the Web browser place a session-level cookie into the visitor's Web browser. This cookie will last as long as the browser is open and, in general, won't be available in subsequent browser sessions. The cookie value can serve as a temporary session ID not only to the browser but also to any application that requests the session cookie from the browser. This request must come from the same Web server (actually, the same domain) that placed the cookie in the first place. Using a transient cookie value as a temporary session ID for both the clickstream and application logging allows a straightforward approach to associating the data from both these sources during post session log processing.

Limitation

Using a transient cookie has the disadvantage that you can't tell when the visitor returns to the site at a later time in a new session with a new transient cookie.

3- Using HTTP's secure sockets layer (SSL)

- Offers an opportunity to track a visitor session
- **Limitations**
  - To track the session, the entire information exchange needs to be in high overhead SSL
  - Each host server must have its own unique security certificate.
  - Visitors are put-off by pop-up certificate boxes.

This offers an opportunity to track a visitor session because it may include a login action by the visitor and the exchange of encryption keys.

Limitations

- The downside to using this method is that to track the session, the entire information exchange needs to be in high overhead SSL, and the visitor may be put off by security advisories that can pop up when certain browsers are used.
- Each host server must have its own unique security certificate.

4- Using session ID Ping-pong

- Maintain visitor state by placing a session ID in a hidden field of each page returned to the visitor.
- Session ID can be returned to the Web server as a query string appended to a subsequent URL.
- **Limitations**
  - Requires a great deal of control over the Web site's page-generation methods
  - Approach breaks down if multiple vendors are supplying content in a single session

If page generation is dynamic, you can try to maintain visitor state by placing a session ID in a hidden field of each page returned to the visitor. This session ID can be returned to the Web server as a query string appended to a subsequent URL.

Limitation

- This method of session tracking requires a great deal of control over the Web site's page-generation methods to ensure that the thread of session ID is not broken. If the visitor clicks on links that don't support this method a single session will appear to be multiple sessions.
- This approach also breaks down if multiple vendors are supplying content in a single session:

5- Using Persistent Cookies

- Establish a persistent cookie in the visitor's PC
- **Limitations**
  - No absolute guarantee that even a persistent cookie will survive.
  - Certain groups of Web sites can agree to store a common ID tag

The Web site may establish a persistent cookie in the visitor's PC that is not deleted by the browser when the session ends.

Limitations

- It's possible that the visitor will have his or her browser set to refuse cookies or may clean out his or her cookie file manually, so there is no absolute guarantee that even a persistent cookie will survive.
- Although any given cookie can be read only by the Web site that caused it to be created, certain groups of Web sites can agree to store a common ID tag that would let these sites combine their separate notions of a visitor session into a super session

Identifying the Visitor

- Identifying a specific visitor who logs onto our site presents some of the most challenging problems
- Web visitors wish to be anonymous.
- If we request a visitor's identity, he or she is likely to lie about it.
- We can't be sure which family member is visiting our site.
- We can't assume that an individual is always at the same computer.

Identifying a specific visitor who logs onto our site presents some of the most challenging problems facing a site designer, Webmaster, or manager of data warehousing for the following reasons:

Web visitors wish to be anonymous. Unlike a bank ATM machine with dedicated access, a user accessing a web site via Internet may have no reason to trust, the Internet, or their PC with personal identification or credit card information.

If we request a visitor's identity, he or she is likely to lie about it. It is believed that when asked their name on an Internet form, men will enter a pseudonym 50 percent of the time and women will use a pseudonym 80 percent of the time.

We can't be sure which family member is visiting our site. If we obtain an identity by association, for instance, from a persistent cookie left during a previous visit, the identification is only for the computer, not for the specific visitor. Any family member or company employee may have been using that particular computer at t hat moment in time

We can't assume that an individual is always at the same computer. Server provided cookies identify a computer, not an individual. If someone accesses the same Web site from an office computer, a home PC, and a laptop computer, a diffe rent Web site cookie is probably put into each machine

Proxy servers

- An HTTP request is not always served from the server specified in a URL.
- Many ISPs make use of proxy servers to reduce Internet traffic.
- Proxy servers can introduce three problems:
    - May deliver outdated content.
    - May satisfy a content request without properly notifying the originating server that the request has been served by the proxy.
    - Web site will not know who made the page request unless a cookie is present.

When a browser makes an HTTP request, that request is not always served from the server specified in a URL. Many ISPs make use of proxy servers to reduce Internet traffic. Proxy servers are used to cache frequently requested content at a location between its intended source and an end visitor. An HTTP request may not even leave the visitor's Pc. It may be satisfied from the browser's local cache of recently accessed objects

Proxy servers can introduce three problems, as illustrated in Figure in next slide

i. A proxy may deliver outdated content. Although Web pages can include tags that tell proxy servers whether or not the content may be cached and when content expires, these tags often are omitted by Webmasters or ignored by proxy servers.

ii. Proxies may satisfy a content request without properly notifying the originating server that the request has been served by the proxy. When a proxy handles a request, convention dictates that it should forward a message that indicates that a proxy response has been made to the intended server, but this is not reliable. As a consequence, the Web warehouse may miss key events that are otherwise required to make sense of the events that comprise a browser/Web site session.

iii. If the visitor has come though a proxy, the Web site will not know who made the page request unless a cookie is present.

Proxey Servers: Problems Illustration

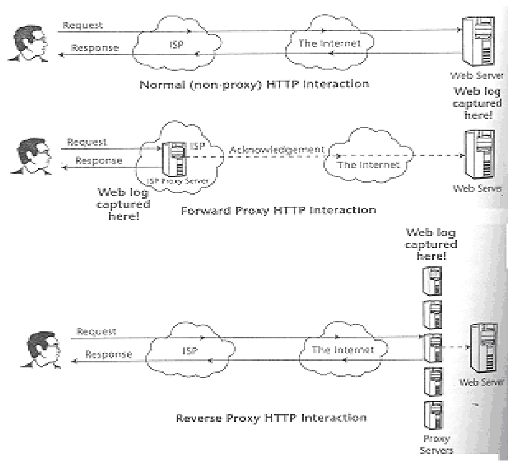


Figure- 40.1: Proxy Server problems

Forward Proxy
The type of proxy we are referring to in this discussion is called a forward proxy. It is outside of our control because it belongs to a networking company or an ISP.

Reverse Proxy
Another type of proxy server, called a reverse proxy, can be placed in front of our enterprise's Web servers to help them offload requests for frequently accessed content. This kind of proxy is entirely within our control and usually presents no impediment to Web warehouse data collection. It should be able to supply the same kind of log information as that produced by a Web server.

Browser caches

- Most browsers store a copy of recently retrieved objects in a local object cache in the PC's file system.
- A visitor may return to a page already in his or her local browser cache
- We can never be certain that we have a full map of the visitor's actions.
- We can attempt to force the browser to always obtain objects from a server rather than from cache
- A similar uncertainty can be introduced when a visitor opens multiple browser windows to the same Web site

Browser caches also introduce uncertainties in our attempts to track all the events that occur during a visitor session. Most browsers store a copy of recently retrieved objects such as HTML pages and images in a local object cache in the PC's file system. If the visitor returns to a page already in his or her local browser cache (for example, by clicking the Back button), no record of this event will be sent to the server, and the event will not be recorded. This means that we can never be certain that we have a full map of the visitor's actions.

As with proxies, we can attempt to force the browser to always obtain objects from a server rather than from cache by including appropriate "No Cache" HTML tags, but we may not choose to do this because of performance or other content-related reasons.

A similar uncertainty can be introduced when a visitor opens multiple browser windows to the same Web site. The visitor may have multiple views of different pages of the site available on his or her PC screen, but there isn't any way for the Web server to know this.

Registration & Login

Most reliable method. Ask users to register and then use unique login ID for visiting Web site.

- How to convince visitors to register? Offer something in return, news, email etc.
- E-Commerce hard to imagine without registration and login.
- Not 100% fool proof. May still have multiple visitors associated with a single login ID (Univ. account).

Hence, the most powerful method of session tracking from Web server is a login similar to an ATM machine. However, how do you make the visitors open an account with you, and then use it. This is not going to be a one way affair, you must offer something in return, for example allowing to read 10% of an article, and the remaining after logging on etc.

Obviously e-commerce is inconceivable without login and account based authentication. However, this is still not fool proof for the straightest reasons too. For example the library of a university may get a paid account for downloading researcher papers, and every active researcher in the university is asking the librarian to get research papers. Or the librarian opens an account with **amazon.com** and searches and buy books for people from different departments.

Some New Challenges

A web warehouse can be implemented with some slight modifications in the development lifecycle, such as:

- Teaming with new partners.
- Beware of slow string functions.
- Large data so min loading.

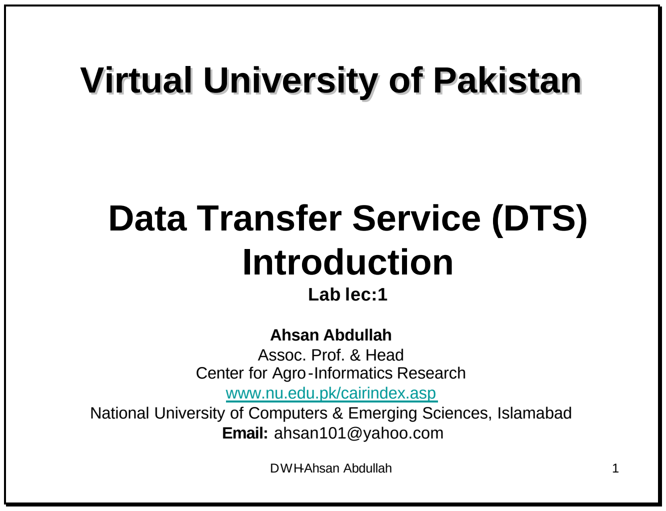
The fact is the clickstream data mart can be implemented just like any other data mart in your data warehouse — with some slight modifications. The data warehouse manager faces the following challenges in the clickstream data mart development life cycle:

- Unlike most data mart implementations, in which primarily DBAs and data administrators own the knowledge of the source data, the data warehouse team will need to work closely with webmasters and programmers. These groups may be the only resources who can supply the crucial information necessary for finding the key elements the business requires.

- The types of substring and instring functions required t o process the Web logs during the ETL process are terribly inefficient in most DBMSs and ETL tools. Because of the daunting size, you must optimize the clickstream ETL process for performance. Inefficient ETL processing will not reach completion in the allocated load window.

- The vast amount of data makes it necessary to limit the revisiting or reloading of logs to an absolute minimum. Therefore, the ETL process must discover, transform, and load new pages, page types, Web servers, and customers during the first pass of the Web log. Additionally, the process must also handle new robots, exclusions, and altered business rules gracefully.

Slide 1



Virtual University of Pakistan

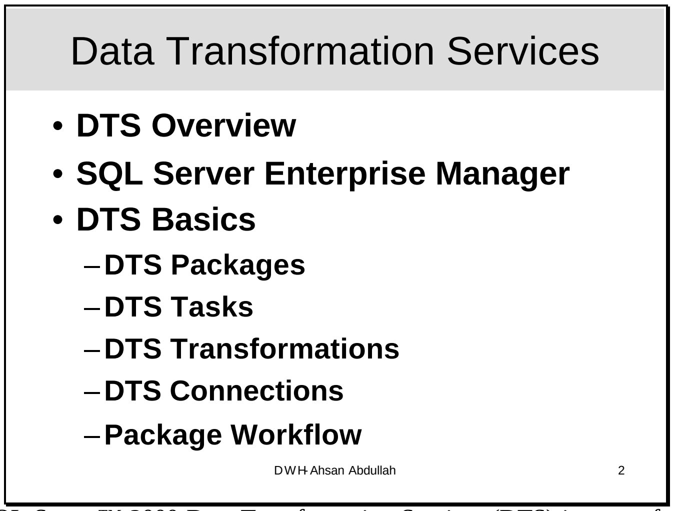
Data Transfer Service (DTS) Introduction

Lab lec:1

Ahsan Abdullah
Assoc. Prof. & Head
Center for Agro-Informatics Research
www.nu.edu.pk/cairindex.asp
National University of Computers & Emerging Sciences, Islamabad
Email: ahsan101@yahoo.com

DWH-Ahsan Abdullah 1

Slide 2



Data Transformation Services

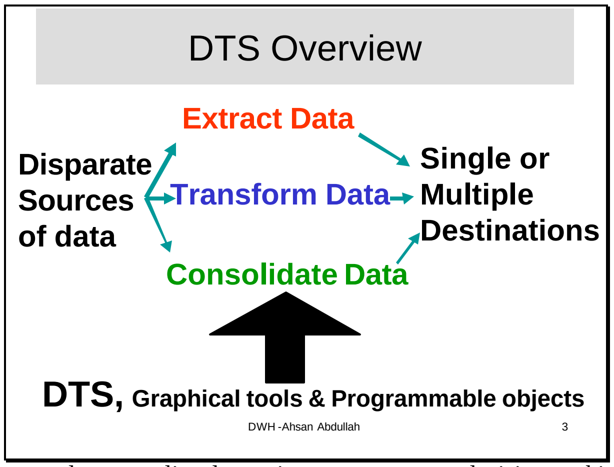
- **DTS Overview**
- **SQL Server Enterprise Manager**
- **DTS Basics**
  - **DTS Packages**
  - **DTS Tasks**
  - **DTS Transformations**
  - **DTS Connections**
  - **Package Workflow**

DWH-Ahsan Abdullah 2

Microsoft® SQL Server™ 2000 Data Transformation Services (DTS) is a set of graphical tools and programmable objects that allow you extract, transform, and consolidate data from disparate sources into single or multiple destinations. SQL Server Enterprise Manager provides an easy access to the tools of DTS.

The purpose of this lecture is to get an understanding of DTS basics, which is necessary to learn the use of DTS tools. These DTS basics describe the capabilities of DTS and summarize the business problems it addresses.

Slide 3



DTS Overview

Extract Data
Disparate Sources of data
Transform Data
Single or Multiple Destinations
Consolidate Data
DTS, Graphical tools & Programmable objects
DWH -Ahsan Abdullah
3

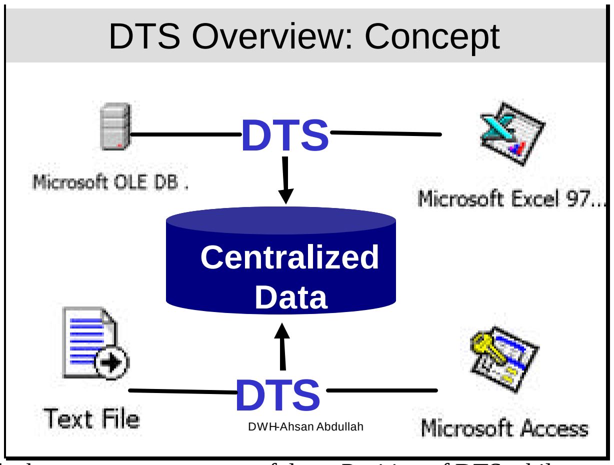
Many organizations need to centralize data to improve corporate decision-making. However, their data may be stored in a variety of formats and in different locations. Data Transformation Services (DTS) address this vital business need by providing a set of tools that let you extract, transform, and consolidate data from disparate sources into single or multiple destinations supported by DTS connectivity.

DTS allows us to connect through any data source or destination that is supported by OLE DB. This wide range of connectivity that is provided by DTS allows us to extract data from wide range of legacy systems. Heterogeneous source systems store data with their local formats and conventions. While consolidating data from variety of sources we need to transform names, addresses, dates etc into a standard format. For example consider a student record management system of a university having four campuses. A campus say 'A' follows convention to store city codes "LHR" for Lahore. An other campus say 'B' stores names of cities "Lahore", campus 'C' stores city names in block letters 'LAHORE', and the last campus 'D' store city names as 'lahore'. When the data from all the four campuses is combined as it is and query is run "How many students belong to 'Lahore'?" We get the answer only from campus B because no other convention for Lahore matches to the one in query.

To combine data from heterogeneous sources with the purpose of some useful analysis requires transformation of data. Transformation brings data in some standard format.

Microsoft SQL Server provides graphical tools to build DTS packages. These tools provide good support for transformations. Complex transformations are achieved through VB Script or Java Script that is loaded in DTS package. Package can also be programmed by using DTS object model instead of using graphical tools but DTS programming is rather complicated.

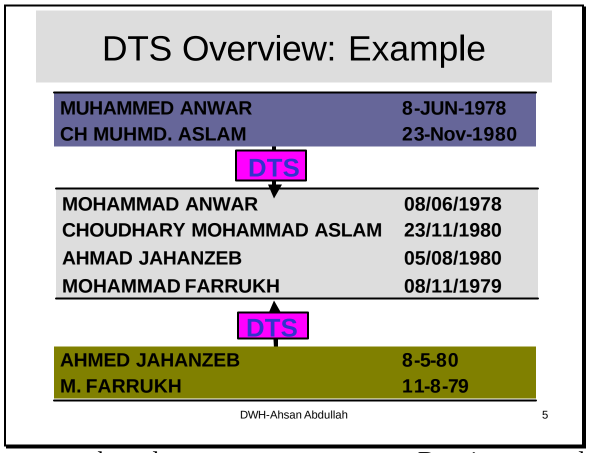
Slide 4



DTS Overview: Concept

The slide shows the heterogeneous sources of data. Position of DTS while consolidating the data into a single source is also clear from the slide. In legacy systems we may come across the text files as a source of data. Microsoft Access is a database management system, maintains data in tables, and columns validate the input to the system. We often find legal values are stored in these sort of data management systems. But when we deal with text files no validation mechanism for input is there. Therefore we may come across illegal and rubbish values in text files. This makes the process of transformation further complicated.

Slide 5



DTS Overview: Example

MUHAMMED ANWAR	8-JUN-1978
CH MUHMD. ASLAM	23-Nov-1980

DTS

MOHAMMAD ANWAR	08/06/1978
CHOUDHARY MOHAMMAD ASLAM	23/11/1980
AHMAD JAHANZEB	05/08/1980
MOHAMMAD FARRUKH	08/11/1979

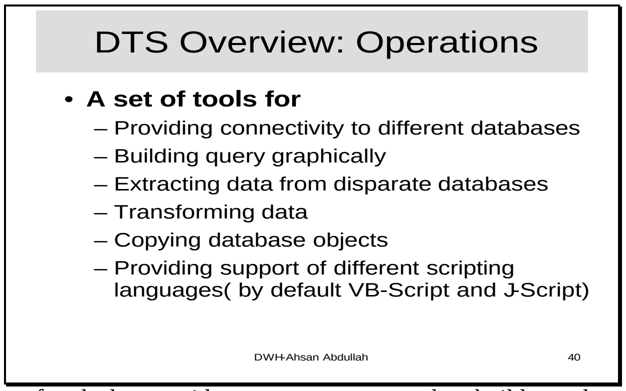
DTS

AHMED JAHANZEB	8-5-80
M. FARRUKH	11-8-79

DWH-Ahsan Abdullah 5

In the this slide we may see three data management systems. Data is extracted from two systems, top and bottom, and is loaded into the standardized system shown in the middle. We may see two transformations over here. First one is name transformation and the other one is date transformation. In the database management system shown at the top we have two names Muhammed Anwer and Choudhary Mohammed Aslam. Whereas in the system shown at the bottom we have two different names Ahmed Jahanzeb and Muhammed Farrukh. Out of four names three names contain Muhammed but with different spellings. Computer can not identify that the word 'Mu hammed' is intended at all the three locations. So while consolidating data

names are transformed to standard spellings of names. Similarly Date formats are different in both source systems and it is standardized in destination system (middle one).
Slide 6



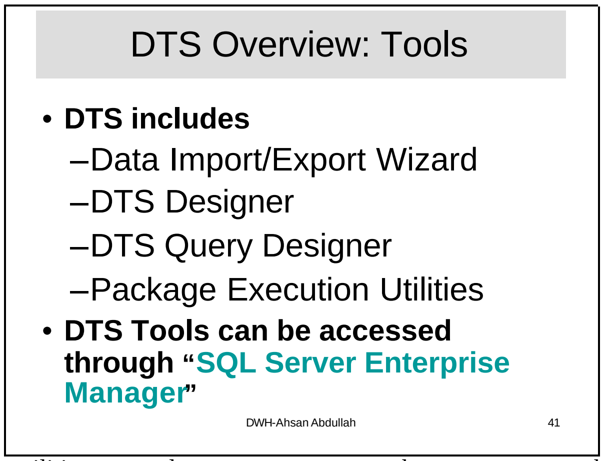
DTS Overview: Operations

- **A set of tools for**
  - Providing connectivity to different databases
  - Building query graphically
  - Extracting data from disparate databases
  - Transforming data
  - Copying database objects
  - Providing support of different scripting languages(by default VB-Script and J-Script)

DWH-Ahsan Abdullah 40

DTS contains a set of tools that provides a very easy approach to build a package and execute it. Writing or building a package through programming is a complex task but DTS tools like DTS Designer and Import/Export Wizard do this entire complex task for user just through a single click of button. Not only package building but query building has also very sophisticated support in DTS tools.

Slide 7



DTS Overview: Tools

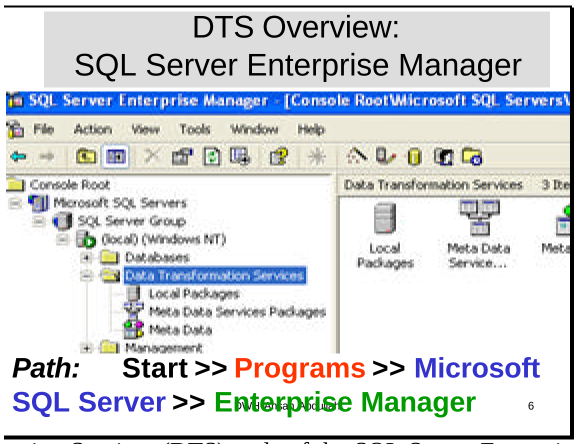
- **DTS includes**
  - Data Import/Export Wizard
  - DTS Designer
  - DTS Query Designer
  - Package Execution Utilities
- **DTS Tools can be accessed through "SQL Server Enterprise Manager"**

DWH-Ahsan Abdullah 41

Package execution utilities are used to run or execute a package, no matter package is designed through the tools provided by DTS or any external tool like Visual Basic. All these tools can be accessed through the SQL Server Enterprise Manager.

Open the node Data Transfer Services in SQL Server Enterprise Manager. Choose the option in which any finished package is saved. Right click the package and get option to execute it.

Slide 8



DTS Overview: SQL Server Enterprise Manager

SQL Server Enterprise Manager - [Console Root\Microsoft SQL Servers\
File Action View Tools Window Help
Console Root
Microsoft SQL Servers
SQL Server Group
(local) (Windows NT)
Databases
Data Transformation Services
Local Packages
Meta Data Services Packages
Meta Data
Management
Data Transformation Services
Local Packages
Meta Data Service...
Meta

Path: **Start >> Programs >> Microsoft SQL Server >> Enterprise Manager**

6

The Data Transformation Services (DTS) node of the SQL Server Enterprise Manager console tree provides facilities for accessing DTS tools, manipulating DTS packages, and accessing package information. You can use these facilities to:

- Open a new package in the DTS Import/Export Wizard or DTS Designer. In DTS Designer, you can select and edit an existing package saved to SQL Server, SQL Server 2000 Meta Data Services, or to a structured storage file.

  Action >> New Package

- Connect to and import meta data from a data source, and display the meta data in the Meta Data node of SQL Server Enterprise Manager.

  Right click Meta Data Services Package and select option import Meta Data

- Open a package template in DTS Designer.

Right click the package that is required to be opened in DTS Designer and select the option open in DTS Designer.

- Display the version history of a package, edit a specific package version in DTS Designer, and delete package versions.

Right click the package and select the option Versions.

- Display and manipulate package log information.

Right click the node "Local packages"/"Meta data services packages" in the tree and select the option view logs.

- Set the properties of DTS Designer

Right-click the Data Transformation Services node and click Properties.

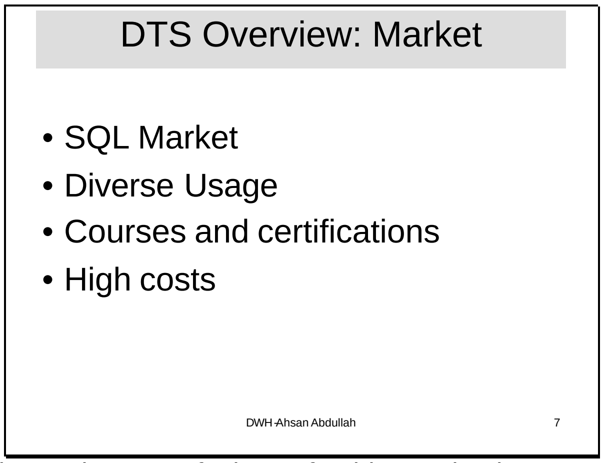
- Execute a package.

Right click the package and click execute

- Schedule a package.

Right click the package and click schedule

Slide 9



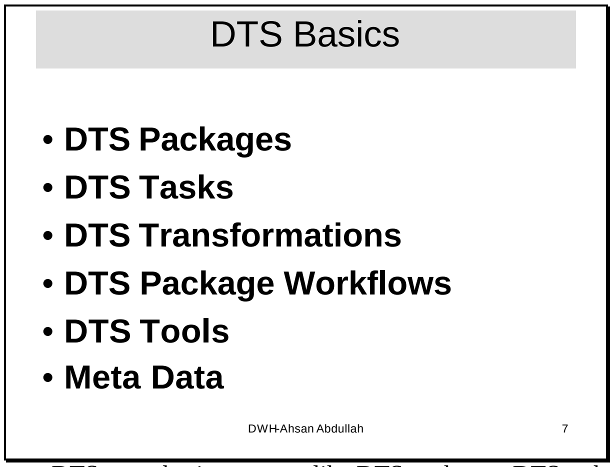
DTS Overview: Market

- SQL Market
- Diverse Usage
- Courses and certifications
- High costs

DWH-Ahsan Abdullah 7

SQL Server is becoming one of Microsoft's biggest businesses, as being used by people from wide spectrum of domains. The scope of the SQL Server is so diverse that whole course can be offered and actually such courses are being offered in developed countries. Microsoft also offers training courses and certifications are expensive enough. For example, "Designing and Implementing OLAP Solutions with MS SQL Server 2000" is a course that provides students with the knowledge and skills necessary to design, implement, and deploy OLAP solutions by using Microsoft SQL Server 2000™ Analysis Services. The importance of the course is well depicted by its cost i.e. $2500+GST

Slide 10


DTS Basics

- **DTS Packages**
- **DTS Tasks**
- **DTS Transformations**
- **DTS Package Workflows**
- **DTS Tools**
- **Meta Data**

DWH-Ahsan Abdullah 7

Before learning to use DTS some basic concepts like DTS packages, DTS tasks, transformations and workflows are important to understand.

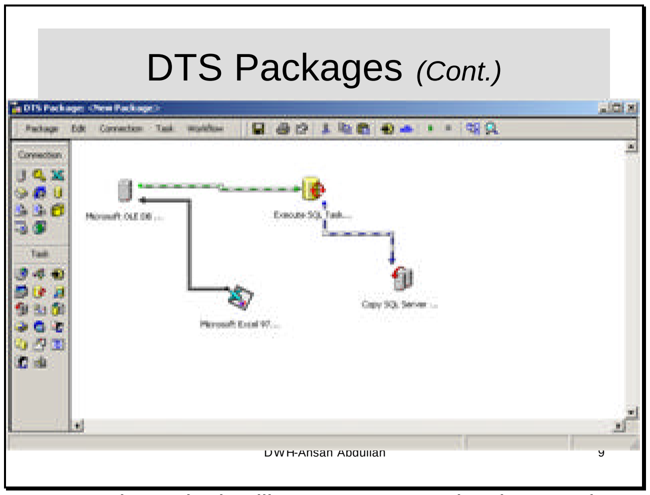
When we want to use computers to perform some particular task through programming, what we do? We write a program in some programming language. Program is a sequence of logical statements that collectively achieve the purpose of the programmer. This analogy is useful in understanding the concept of package and tasks in DTS. DTS package is exactly like a computer program. Like a computer program DTS package is also prepared to achieve some goal. Computer program contains set of instructions whereas DTS package contains set of tasks. Tasks are logically related to each other. When a computer program is run, some instructions are executed in sequence and some in parallel. Likewise when a DTS package is run some tasks are performed in sequence and some in parallel. The intended goal of a computer program is achieved when all instructions are successfully executed. Similarly the intended goal of a package is achieved when all tasks are successfully accomplished.

DTS task is a unit of work in a package. Tasks can be establishment of connection to source and destination databases, extraction of data from the source, transformation of data, loading of data to the destination, generation of error messages and emails etc.

In real world systems when we talk about heterogeneous sources of data there arise a lot of complicated issues. Heterogeneous systems contain data with different storage conventions, different storage formats, different technologies, and different designs etc. Power of DTS lies in extracting the data from these heterogeneous sources, transforming to some standard format and convention, and finally load data to some different system with totally different parameters like technology, design etc. Microsoft SQL Server provides user-friendly tools to develop DTS Packages. Through graphical editor/ designer or wizards we can put together set of tasks in a package. Order or sequence in which the tasks are required to be performed can be set through conditions like “On success of task A task B should be performed otherwise task C should be performed.” This order or sequence of execution is called Workflow of a package.

In this lecture we will see these concepts in detail and in subsequent lectures we will develop packages and practically get into the use of DTS functionalities.

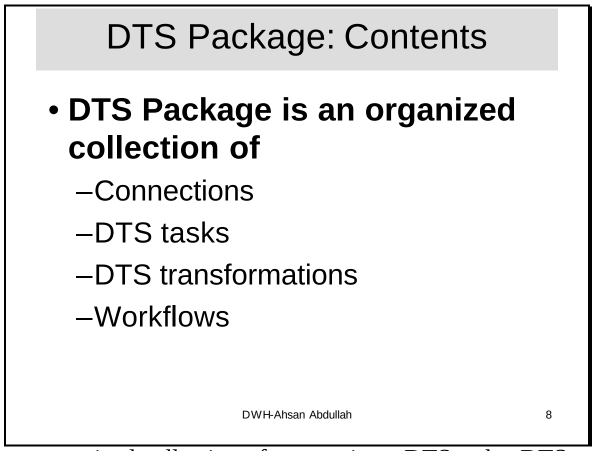
Slide 11


DTS Packages *(Cont.)*

DWH-Ahsan Abdullah 9

Slide shows how a package looks like. We can only view package as a form of graphical objects as shown in the slide. Here two connections are established. “Microsoft OLEDB Driver” and “Microsoft Excel 97” are connections. Blac k link between two connections is transformation task. “Execute SQL” and “Copy SQL Server” both are tasks. Green and blue links are workflows. Green link shows ***‘On the Success of’*** *i.e.* on the success of Connection establishment execute task execute SQL.Blue link shows ***‘On the Failure of’*** on the failure of the previous task execute another task Copy SQL Server objects.

Slide 12


DTS Package: Contents

- **DTS Package is an organized collection of**
  - –Connections
  - –DTS tasks
  - –DTS transformations
  - –Workflows

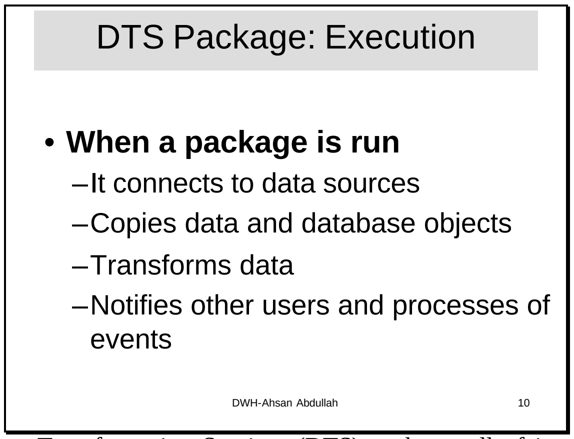
DWH-Ahsan Abdullah 8

A DTS package is an organized collection of connections, DTS tasks, DTS transformations, and workflow constraints assembled either with a DTS tool or programmatically and saved to

Microsoft® SQL Server™, SQL Server 2000 Meta Data Services, a structured storage file, or a Microsoft Visual Basic® file.

Each package contains one or more steps that are executed sequentially or in parallel when the package is run. When executed, the package connects to the correct data sources, copies data and database objects, transforms data, and notifies other users or processes of events.

Slide 13



DTS Package: Execution

- **When a package is run**
  - It connects to data sources
  - Copies data and database objects
  - Transforms data
  - Notifies other users and processes of events

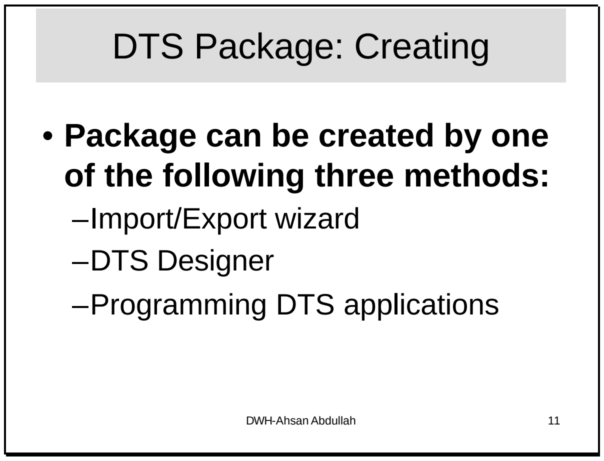
DWH-Ahsan Abdullah 10

When we run a Data Transformation Services (DTS) package, all of its connections, tasks, transformations, and scripting code are executed in the sequence described by the package workflow.

We can execute a package from:

- Within a DTS tool.
- SQL Server Enterprise Manager.
- Package execution utilities.

Slide1 4



DTS Package: Creating

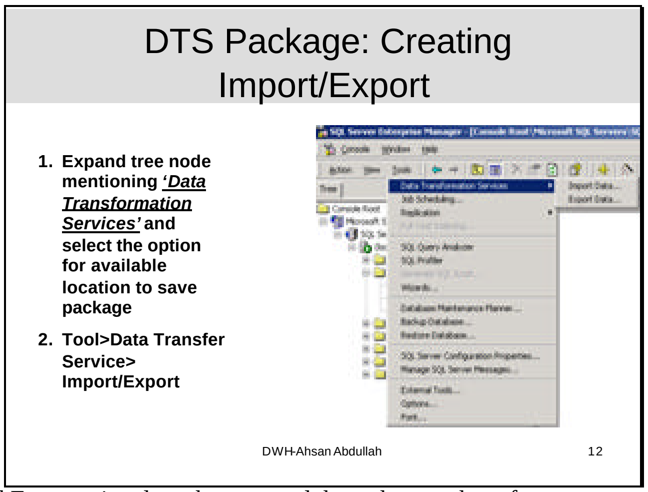
- **Package can be created by one of the following three methods:**
  - Import/Export wizard
  - DTS Designer
  - Programming DTS applications

DWH-Ahsan Abdullah 11

Microsoft SQL Server provides a good support for the tools that are helpful in building a package. Import/Export Wizard and DTS Designer both are the graphical methods of building a package. Both tools provide support to run the package also. Building a package means putting all the tasks that are supposed to be performed in a particular package together and setting their order of execution or defining workflow. Whereas when we actually run a package all the tasks are actually performed.

Programming DTS applications without the help of these user-friendly tools is a difficult task. Packages can be programmed using some external tool like Visual Studio in VC++ or VB. Such a programming requires deep understanding of DTS object model.

Slide 15



DTS Package: Creating Import/Export

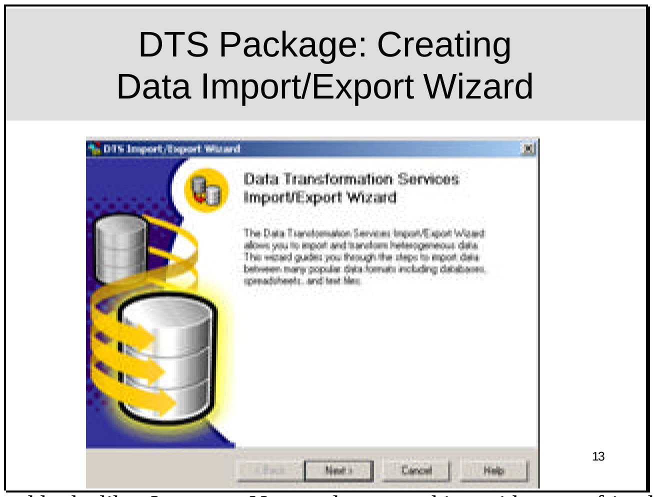
1. **Expand tree node mentioning *<u>'Data Transformation Services'</u>* and select the option for available location to save package**
2. **Tool>Data Transfer Service> Import/Export**

DWH-Ahsan Abdullah 12

Data Import and Export wizard can be accessed through a number of ways.

1. Start > Programs> Microsoft SQL Server> Data Import/Export Wizard
2. Through SQL Server Enterprise Manager
   a. In SQL Server Enterprise Manager we can see a Tree view of SQL Server objects and services. Expand Tree, select the node "Data Transformation Services". We can see two options (discussed earlier) to store Package. Select any one of them (local OR SQL Server Meta Data Services). Then Click Tools > Data Transformation Services > Import / Export Data.
   b. After Expanding Data Transformation Services node we can click on tool bar to launch the wizard

Slide 16



DTS Package: Creating Data Import/Export Wizard

DTS Import/Export Wizard

Data Transformation Services Import/Export Wizard

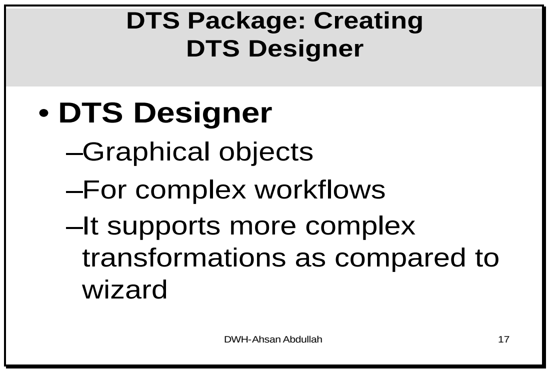
The Data Transformation Services Import/Export Wizard allows you to import and transform heterogeneous data. This wizard guides you through the steps to import data between many popular data formats including databases, spreadsheets, and text files.

13

This is how wizard looks like. Just press Next and start working with a user friendly wizard.

An easy-to-use tool that guides you, a step at a time, through the process of creating a DTS package. It is recommended for simple data transformation or data movement solutions (for example, importing tabular data into a SQL Server 2000 database). It provides limited support for transformations.

Slide 17



DTS Package: Creating DTS Designer

- **DTS Designer**
  - –Graphical objects
  - –For complex workflows
  - –It supports more complex transformations as compared to wizard

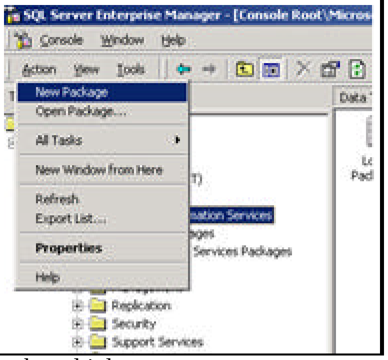
DWH-Ahsan Abdullah 17

DTS designer is an application that uses graphical objects to help you build packages containing complex workflows. DTS Designer includes a set of model DTS Package Templates, each designed for a specific solution that you can copy and customize for your own installation. It is recommended for sophisticated data transformation solutions requiring multiple connections, complex workflows, and event-driven logic. DTS package templates are geared toward new users who are learning about DTS Designer or more experienced users who want assistance setting up specific DTS functionalities (for example, data driven queries).

Slide 18

DTS Package: Creating
DTS Designer Console

1. Expand tree node mentioning ***'Data Transformation Services'*** and select the option for available location to save package
2. Action>New Package

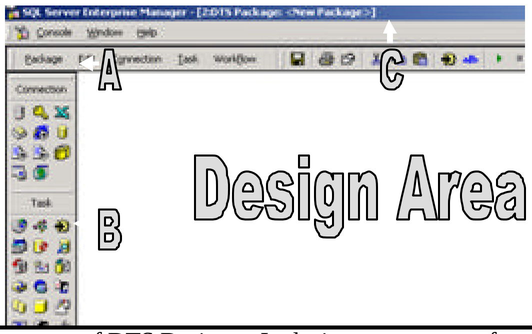


DTS Designer can also be accessed through multiple ways.

1. Whenever a saved package is opened by double click or through right click, it is opened in DTS Designer
2. SQL Server Enterprise Manager can also be used to access DTS Designer
   a. After Expanding Data Transformation Services node select Action > New Package
   b. After Expanding Data Transformation Services node select ✳ on toolbar to access DTS Designer

Slide 19

DTS Package: Creating
DTS Designer Environment


SQL Server Enterprise Manager - [2:DTS Package: <New Package>]
Console Window Help
Package Edit Connection Task Workflow
Connection
Task
A
B
C
Design Area

The slide shows environment of DTS Designer. In designer we can see four windows

A. Connection toolbar
B. Task toolbar
C. General toolbar
D. Design Area

A. **Connection toolbar**

Connection toolbar shows all available connections in the form of icons or symbols. All OLE DB supported connections are available. To establis h a new connection just click the correct icon and drag to design area. Then set properties to your connection. In case of any difficulty in identifying the connection icon, click on Connection on Menu bar just above the connection toolbar.

B. **Task Toolbar**

Tasks toolbar shows icons for all tasks that are supported by DTS. For example ⚙ is used to set transformation task. This also works as drag and drop. DTS Designer is very friendly to use as it guides user about what to do after picking a certain option. For new users who do not recognize the tasks through icons, in the top menu bar 'Task' is available.

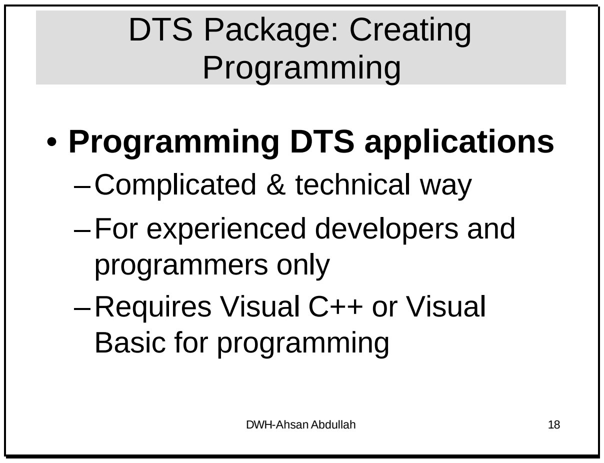
C. **General Toolbar**

This toolbar provides general functionality like saving a package, executing a package. ▶ is used to execute a package

D. **Design Area**

Design Area is used to design a package through the objects available in the tool bars.

Slide 20



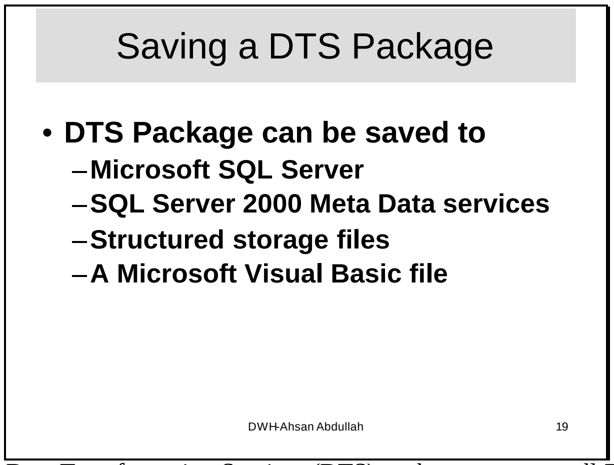
DTS Package: Creating Programming

- **Programming DTS applications**
  - Complicated & technical way
  - For experienced developers and programmers only
  - Requires Visual C++ or Visual Basic for programming

DWH-Ahsan Abdullah 18

Programming applications that you can use to write and compile a DTS package either in Microsoft Visual Basic® or Microsoft Visual C++®. It is recommended for developers who want to access the DTS object model directly and exert a fine degree of control over package operations. Packages created programmatically can be opened and further customized in DTS Designer. In addition, packages created in the DTS Import/Export Wizard or DTS Designer can be saved as a Visual Basic program and then opened and further customized in a development environment such as Microsoft Visual Studio®.

Slide 19



Saving a DTS Package

- DTS Package can be saved to
  - Microsoft SQL Server
  - SQL Server 2000 Meta Data services
  - Structured storage files
  - A Microsoft Visual Basic file

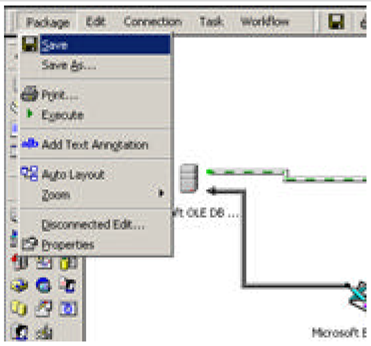
DWH-Ahsan Abdullah 19

When you save a Data Transformation Services (DTS) package, you save all DTS connections, DTS tasks, DTS transformations, and workflow steps and preserve the graphical layout of these objects on the DTS Designer design

While saving a package we get different options as destination location for the package. Package can be saved to Microsoft SQL server. Another option to save a package is SQL Server Meta Data Services. The advantage which we get when we store our package to SQL Server 2000 Meta Data Services is that we may maintain meta data information of the databases involved in the packages and we may keep version information of each package. Furthermore package can be stored in a structured file and Microsoft visual basic file.

Slide 20

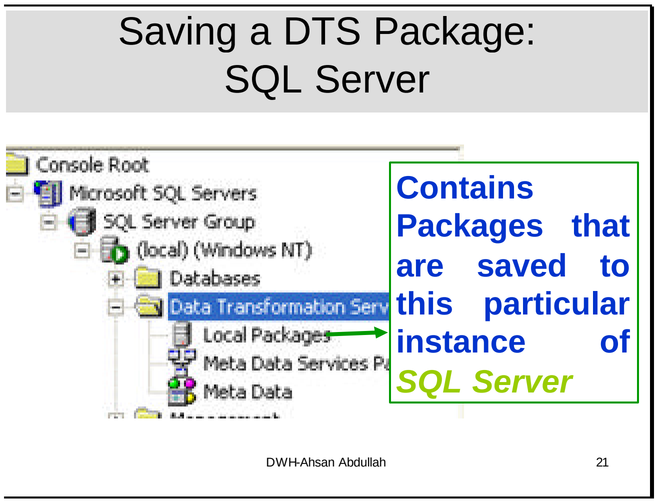
Saving a DTS Package: Illustration



DWH-Ahsan Abdullah 20

This slide illustrate the package saving process.

Slide 21



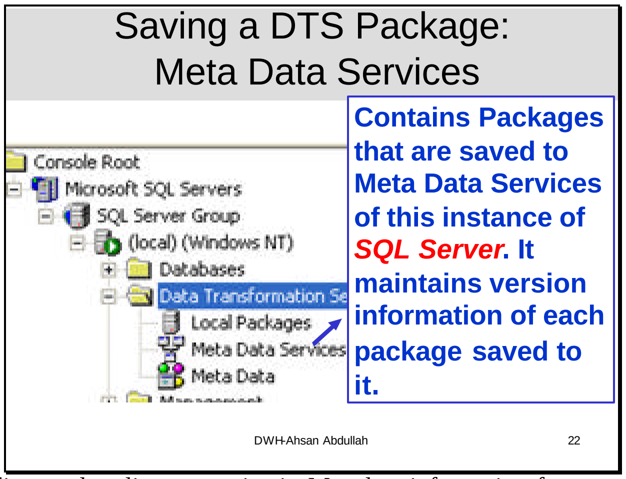
Saving a DTS Package: SQL Server

Contains Packages that are saved to this particular instance of *SQL Server*

DWH-Ahsan Abdullah 21

Data Transformation Services node of SQL Server Enterprise Manager contains three options to locate the package saved earlier. The first option is local Packages. These are the packages that are saved to this particular instance of SQL Server. Microsoft SQL Server may have multiple instances on each machine or over a Network. Local packages are those that are saved to this particular instance of SQL Server.

Slide 22



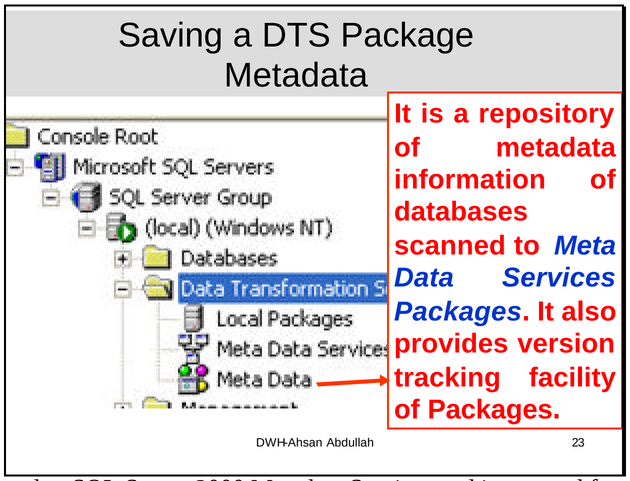
Saving a DTS Package: Meta Data Services

Contains Packages that are saved to Meta Data Services of this instance of ***SQL Server*. It maintains version information of each package saved to it.**

DWH-Ahsan Abdullah 22

As it has been discussed earlier, to maintain Metadata information for a package or version information of a package, it may be stored to Meta Data Services. This node contains all those packages that are saved to SQL Server 2000 Meta Data Services.

Slide 23



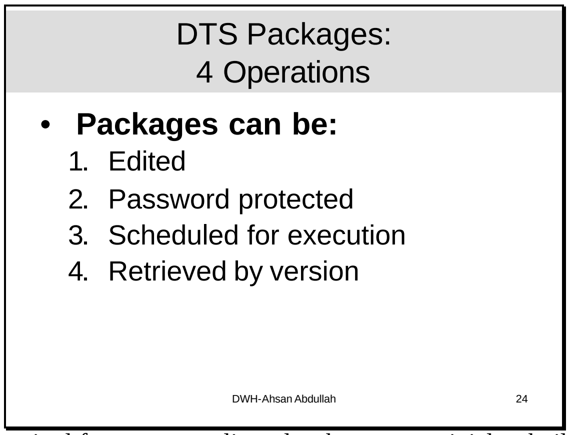
Saving a DTS Package Metadata

It is a repository of metadata information of databases scanned to *Meta Data Services Packages*. It also provides version tracking facility of Packages.

DWH-Ahsan Abdullah 23

If a package is saved to SQL Server 2000 Metadata Services and is scanned for metadata than its meta data information is maintained in a repository " Meta Data ", that can be found as third option under node Data Transformation Services.

Slide 24



DTS Packages: 4 Operations

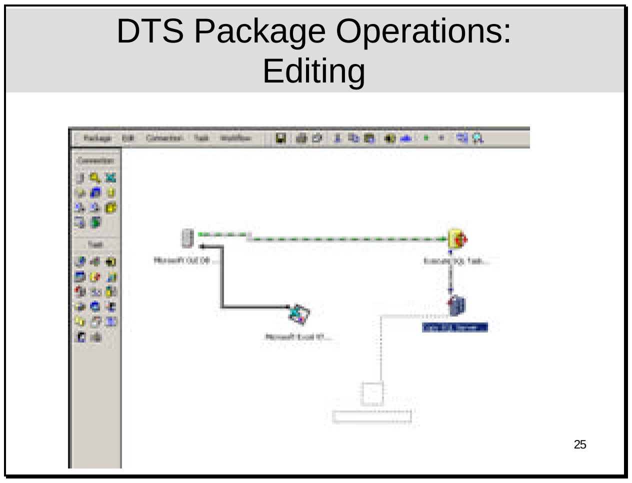
- **Packages can be:**
  1. Edited
  2. Password protected
  3. Scheduled for execution
  4. Retrieved by version

DWH-Ahsan Abdullah 24

Packages that are required for very complicated tasks are not trivial to build. To develop such packages, DTS Designer or programming tools are used. Once such packages are built they are saved for further use. We may edit these packages later on. For editing purposes either DTS designer or programming is used. After edit a package we may keep both packages that is package before editing and package after editing as two different versions of same package. To maintain version information packages are saved in "SQL Server 2000 Meta Data Services".

Tasks in the package require access to database objects, when package is executed. Packages can be protected through passwords. When a package is built it is not necessary to execute it immediately. We may schedule package to be executed any time later on. We may prepare a package that is executed after definite intervals. For example we want to update our dataware house every night at 12:00 o' clock, what will we do? We will write a package to update dataware house and schedule it to run at 12:00 o' clock every night.

Slide 25

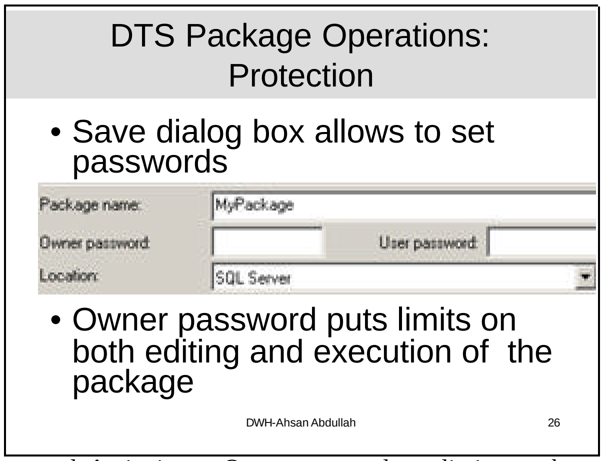


DTS Package Operations: Editing

25

Double click a package to open in designer. Drag and drop objects to edit a package. Designer is the easiest way of editing a package. Even the packages that are created through wizards and are saved, can be edited through designer.

Slide 26



DTS Package Operations: Protection

- Save dialog box allows to set passwords

Package name:	MyPackage		
Owner password:		User password:	
Location:	SQL Server		

- Owner password puts limits on both editing and execution of the package

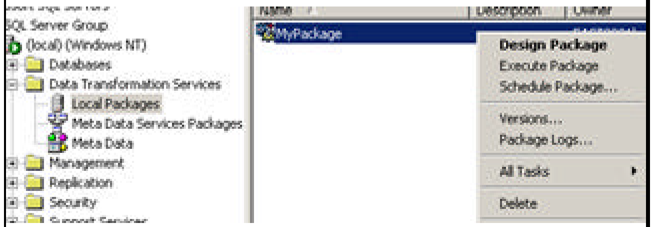
DWH-Ahsan Abdullah 26

Enter an Owner password. Assigning an Owner password puts limits on who can both edit and run the package.

Enter a User password. Assigning a User password puts limits only on who can edit the package. If you create a User password, you must also create an Owner password.

Slide 27

DTS Package Operations: Scheduling/Execution



- Right click any saved package to schedule or execute it

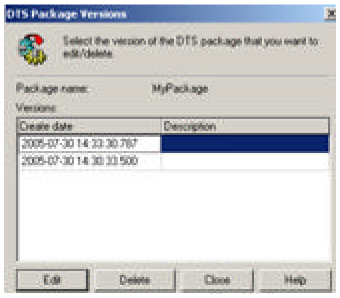
DWH-Ahsan Abdullah 27

To schedule a package or to execute a package, first right click the package and then select the required option.

Slide 28

DTS Package Operations: Versioning

- Right click any saved package to view its version information



Select the version of the DTS package that you want to edit/delete.

Package name: MyPackage

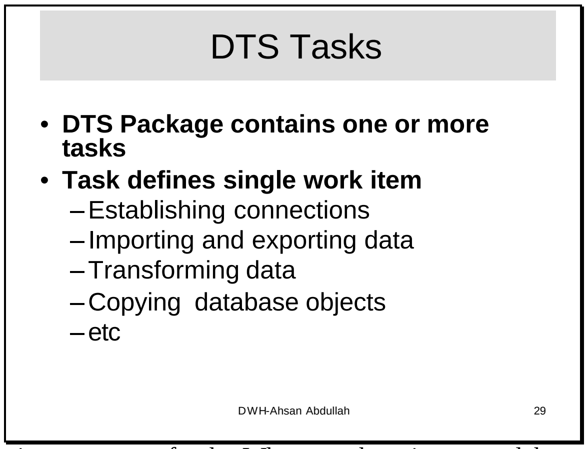
Versions:

Create date	Description
2005-07-30 14:33:30.787	
2005-07-30 14:30:33.500	

28

If we want to get version information of a package we can see it by right clicking the package and selecting version information. First column contains creation date and the other column contains the description about changes if it is saved with the package.

Slide 29



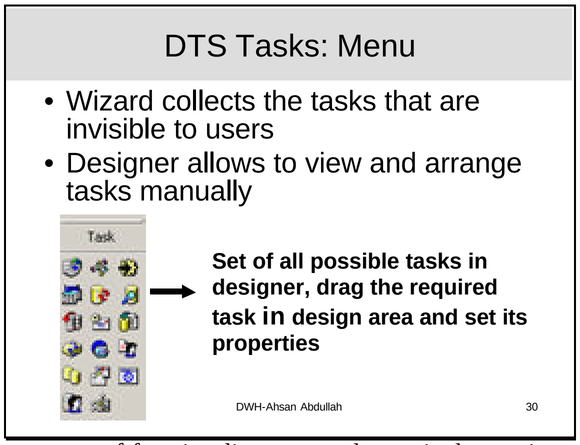
DTS Tasks

- **DTS Package contains one or more tasks**
- **Task defines single work item**
  - Establishing connections
  - Importing and exporting data
  - Transforming data
  - Copying database objects
  - etc

DWH-Ahsan Abdullah 29

DTSPackages contain a sequence of tasks. When a package is executed these tasks are performed in sequence or in parallel. These tasks are the single work item in a package. Tasks can be establishing connections, extraction of data from sources, transformations applied on data, loading data to destination, generation of automated email messages to administrator in case of some problem during the package execution..

Slide 30



DTS Tasks: Menu

- Wizard collects the tasks that are invisible to users
- Designer allows to view and arrange tasks manually

Set of all possible tasks in designer, drag the required task in design area and set its properties

DWH-Ahsan Abdullah 30

A DTS task is a discrete set of functionality, executed as a single step in a package. Each task defines a work item to be performed as part of the data movement and data transformation process, or as a job to be executed.

DTS supplies a number of tasks that are part of the DTS object model and can be accessed graphically, through DTS Designer, or programmatically. These tasks, which can be configured individually, cover a wide variety of data copying, data transformation, and notification situations. For example:

Importing and exporting data. DTS can import data from a text file or an OLE DB data source (for example, a Microsoft Access 2000 database) into SQL Server. Alternatively, data can be exported from SQL Server to an OLE DB data destination (for example, a Microsoft Excel 2000 spreadsheet). DTS also allows high-speed data loading from text files into SQL Server tables.

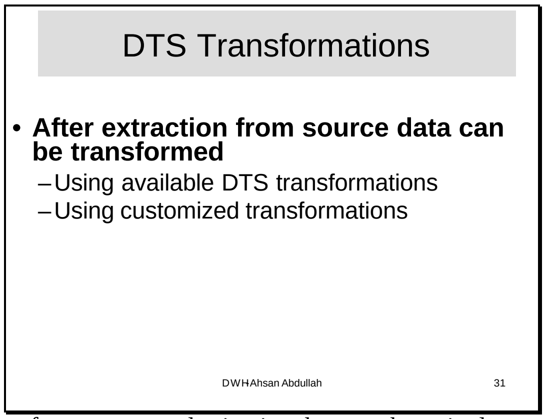
Transforming data. DTS Designer includes a Transform Data task that allows you to select data from a data source connection, map the columns of data to a set of transformations, and send the transformed data to a destination connection. DTS Designer also includes a Data Driven Query task that allows you to map data to parameterized queries.

Copying database objects. With DTS, you can transfer indexes, views, logins, stored procedures, triggers, rules, defaults, constraints, and user-defined data types in addition to the data.

In addition, you can generate the scripts to copy the database objects. Sending and receiving messages to and from other users and packages. DTS includes a Send Mail task that allows you to send an e-mail if a package step succeeds or fails. DTS also includes an Execute Package task that allows one package to run another as a package step, and a Message Queue task that allows you to use Message Queuing to send and receive messages between packages.

Executing a set of Transact-SQL statements or Microsoft ActiveX® scripts against a data source. The Execute SQL and ActiveX Script tasks allow you to write your own SQL statements and scripting code and execute them as a step in a package workflow.

Slide 31



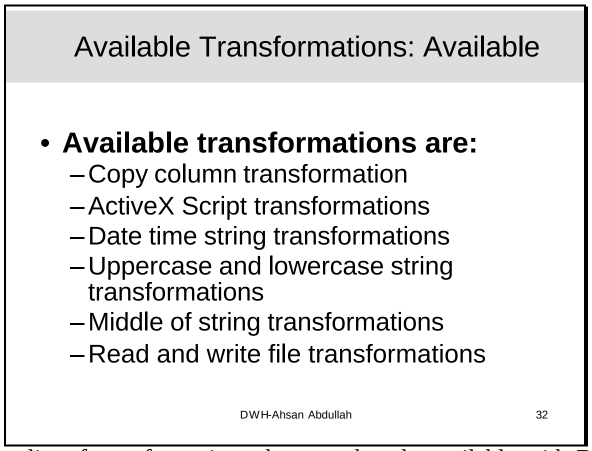
DTS Transformations

- **After extraction from source data can be transformed**
  - Using available DTS transformations
  - Using customized transformations

DWH-Ahsan Abdullah 31

While transferring data from source to destination that may be a single source of truth, data may require to be transformed. Power of DTS tools lies in the support of data transformations. Some transformations are already available with DTS tools and customized transformations can be performed through VB Script or Java Script.

Slide 32



Available Transformations: Available

- **Available transformations are:**
  - Copy column transformation
  - ActiveX Script transformations
  - Date time string transformations
  - Uppercase and lowercase string transformations
  - Middle of string transformations
  - Read and write file transformations

DWH-Ahsan Abdullah 32

The slide shows the list of transformations that are already available with DTS tools i.e. DTS Designer and DTS import/export wizard. Wizard has a support of two transformations out of six shown over here:

- Copy column transformation
- Active-X script transformation

The rest four are accessed through DTS designer and scripts.

Copy Column Transformation: Describes the transformation used to copy source data to the destination.

ActiveX Script Transformation: Explains how to use Microsoft ActiveX® scripts to define column-level transformations.

Date Time String Transformation: Describes the transformation used to convert a source date into a new destination format.

Uppercase String Transformation: Describes the transformation used to convert a string into uppercase characters.

Lowercase String Transformation: Describes the transformation used to convert a string into lowercase characters.

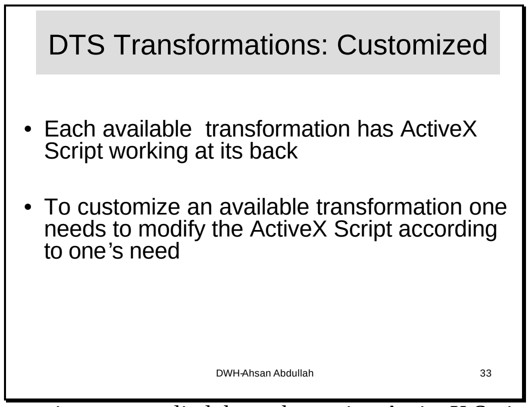
Middle of String Transformation: Describes the transformation used to extract a substring from a source and optionally change its case or trim white space before placing the result in the destination.

Trim String Transformation: Describes the transformation used to remove leading, trailing, or embedded white space from a source string and place the (optionally case-shifted) result in the destination.

Read File Transformation: Describes the transformation used to copy the contents of a file specified by a source column to a destination column.

Write File Transformation: Describes the transformation that creates a new data file for each file named in a source column and initializes the contents of each file from data in a second source column.

Slide 33



DTS Transformations: Customized

- Each available transformation has ActiveX Script working at its back
- To customize an available transformation one needs to modify the ActiveX Script according to one's need

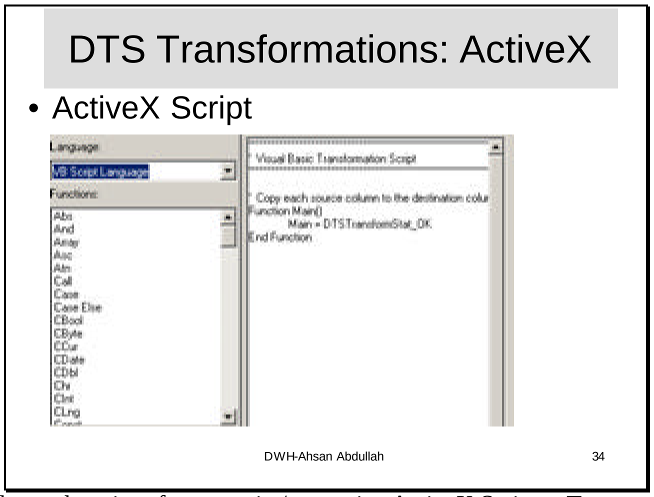
DWH-Ahsan Abdullah 33

In SQL Server, transformations are applied through running ActiveX Scripts. When we apply an available transformation, tools in SQL Server generate ActiveX Script automatically. This auto generated script can be modified or customized according to our needs. Customized transformations are those in which we customize the auto generated ActiveX Scripts to fulfill our particular need.

For Example:

An available transformation can transform Saad Munir Rao to SAAD MUNIR RAO but if we want to transform it as S. M. Rao then we need to customize the transformation.

Slide 34



DTS Transformations: ActiveX

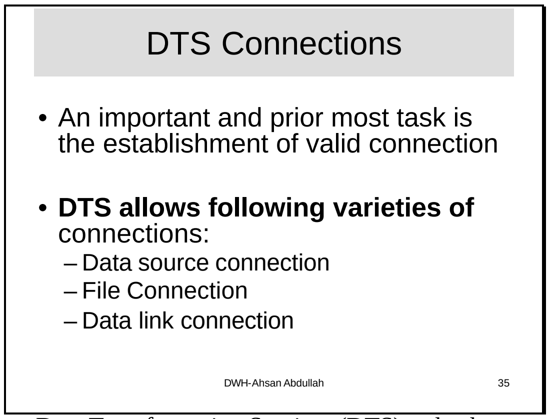
- ActiveX Script

```
' Visual Basic Transformation Script

' Copy each source column to the destination colu
Function Main()
        Main = DTSTransformStat_OK
End Function
```

DWH-Ahsan Abdullah 34

Designer provides such an interface to write/customize ActiveX Scripts. To access it we will see it later on.

Slide 35



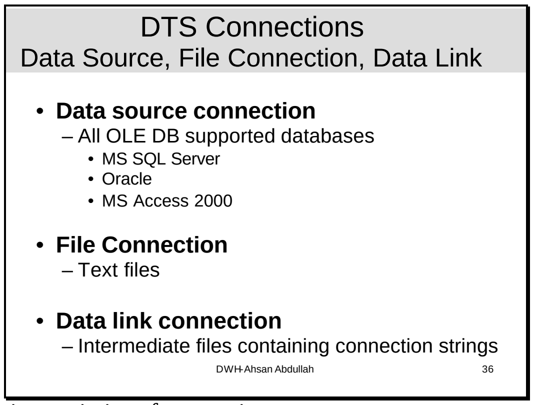
DTS Connections

- An important and prior most task is the establishment of valid connection
- **DTS allows following varieties of** connections:
  - Data source connection
  - File Connection
  - Data link connection

DWH-Ahsan Abdullah 35

To successfully execute Data Transformation Services (DTS) tasks that copy and transform data, a DTS package must establish valid connections to its source and destination data and to any additional data sources (for example, lookup tables). Because of its OLE DB architecture, DTS allows connections to data stored in a wide variety of OLE DB-compliant formats. In addition, DTS packages usually can connect to data in custom or nonstandard formats if OLE DB providers are available for those data sources and if you use Microsoft® Data Link files to configure those connections

Slide 36



DTS Connections
Data Source, File Connection, Data Link

- **Data source connection**
  - All OLE DB supported databases
    - MS SQL Server
    - Oracle
    - MS Access 2000
- **File Connection**
  - Text files
- **Data link connection**
  - Intermediate files containing connection strings

DWH-Ahsan Abdullah 36

DTS allows the following varieties of connections:
A data source connection. These are connections to: standard databases such as Microsoft SQL Server™ 2000, Microsoft Access 2000, Oracle, dBase, Paradox; OLE DB connections to ODBC data sources; Microsoft Excel 2000 spreadsheet data; HTML sources; and other OLE DB providers.

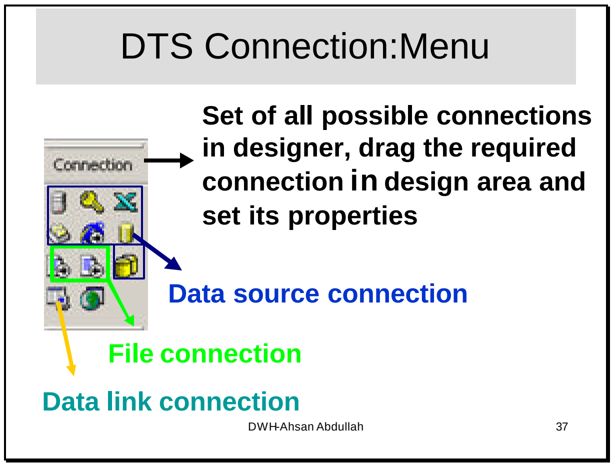
A file connection. DTS provides additional support for text files. When specifying a text file connection, you specify the format of the file. For example:

- Whether a text file is in delimited or fixed field format.
  Whether the text file is in a Unicode or an ANSI format.
- The row delimiter and column delimiter if the text file is in fixed field format.

- The text qualifier.
  Whether the first row contains column names.

A data link connection. These are connections in which an intermediate file outside of SQL Server stores the connection string.

Slide 37

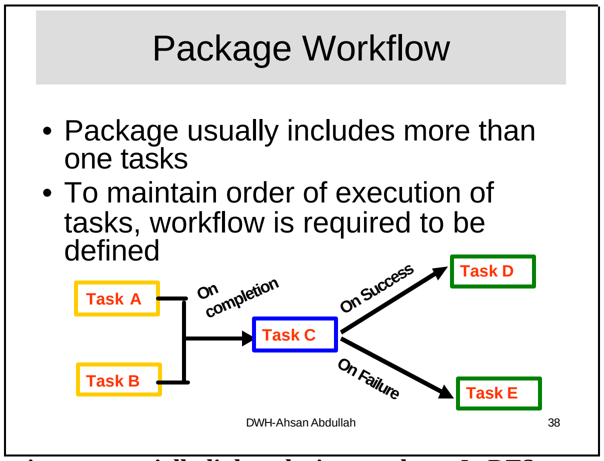


DTS Connection:Menu

Set of all possible connections in designer, drag the required connection in design area and set its properties

DWH-Ahsan Abdullah 37

Slide 38



Package Workflow

- Package usually includes more than one tasks
- To maintain order of execution of tasks, workflow is required to be defined

DWH-Ahsan Abdullah 38

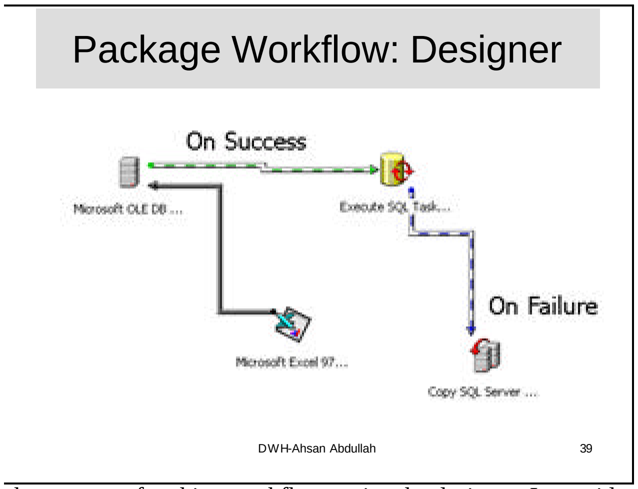
Precedence constraints sequentially link tasks in a package. In DTS, you can use three types of precedence constraints, which can be accessed either through DTS Designer or programmatically:

Unconditional. If you want Task 2 to wait until Task 1 completes, regardless of the outcome, link Task 1 to Task 2 with an *unconditional precedence* constraint.

On Success. If you want Task 2 to wait until Task 1 has successfully completed, link Task 1 to Task 2 with an *On Success precedence* constraint.

On Failure. If you want Task 2 to begin execution only if Task 1 fails to execute successfully, link Task 1 to Task 2 with an *On Failure precedence* constraint. If you want to run an alternative branch of the workflow when an error is encountered, use this constraint.

Slide 39



Package Workflow: Designer

DWH-Ahsan Abdullah 39

This slide shows the process of making workflows using the designer. It provides a graphical interface making the workflow management very easy

Lab Lect-2
Lab Data Set

In previous lecture I gave you an overview of the tool to be used for the lab i.e. Data Transformation Services (DTS), MS SQL Server. Now keeping in view the real issue of data acquisition, we will provide you with a simulated data set, so as to make you ready to start exploring the tool. The data is for a multi-campus university having campuses in four major cities. We discussed the details of such a university in Lect-6 of the course i.e. normalization. Each of the campus has its own conventions and norms regarding storing Student information.

Multi -Campus University

- **University has four campuses situated at:**
  - Lahore
  - Karachi
  - Islamabad
  - Peshawar
- **University Head Office in Islamabad**

Data warehouse is a single source of truth. We have to put all data from different data sources (campuses) at one place in some standard form. The task is not trivial. Different sources of data have a lot of inherent issues of ETL. High level steps given in the slide give just an overview of the task. First of all, we have to identify the source systems. It is quite possible that each campus uses different database systems or same organization at different geographical locations uses different database management systems. To put data into a single source, after extracting from such diverse sources, requires powerful tools especially designed to fulfill the requirements of ETL. We will use Microsoft SQL Server DTS which is a user friendly graphical tool and makes such a complex task doable by some practice. After identification of source systems, it is necessary to study the issues that must be considered before putting all the data together at a single location. Microsoft SQL Server provides a powerful support to perform Extract, Transform and Load (ETL) data from source systems to destination system. Finally certain steps are performed to check and improve quality of data.

In this lab lecture we will look into the data for each of the campuses in detail. This would lead us to identify the core issues that are needed to be taken care of before extracting data from these diverse sources into a single destination.

Degree Programs

- At each campus university has two degree programs:
  - BS
  - MS
- University started its BS degree program in year 1994 and MS degree program in year 2001

Our Example University offers undergraduate and graduate degrees in all of its campuses. The undergraduate degrees were started in year 1994 and graduate degrees were started in year 2001.

Disciplines for BS

- **Four disciplines at BS level**
  - Computer Science (CS)
  - Computer Engineering (CE)
  - System Engineering (SE)
  - Telecommunication (TC)
- All campuses offer these four disciplines

The slide is self explanatory.

Disciplines for MS

- **Four disciplines at MS level**
  - Computer Science (MS-CS)
  - Software Project Mgmt. (MS-SPM)
  - Networking (MS-NW)
  - Telecommunication (MS-TC)
- Lahore & Karachi campuses offer all the four disciplines
- Islamabad offers MS-CS & MS-SPM
- Peshawar offers MS-CS & MS-TC

The slide is self explanatory.

The need

- Four campuses of the University maintain their students record locally
- No standardized way of record management
- Standardized reporting is difficult and time consuming.
- No centralized repository of data
- Head Office wants a central data repository for decision support i.e. a DWH
- We will study the record management at each campus
- In this lecture, we will collect data from each campus and figure out the issues

As mentioned earlier, our example university has multiple campuses and each campus independently maintains its student records without any meaningful level of coordination. There is no any standardized record management system or agreement among these campuses. Each of the campuses uses its own student record management practices independent of the other campuses. The head office of the university now wants to consolidate the student records from all of the four campuses into a central repository for decision support. Thus they are planning for a DWH.

Students Record Keeping & Mgmt.

- One by one we discuss the record management system specific to each campus of the University

  1. Lahore
  2. Karachi
  3. Islamabad
  4. Peshawar

In real life when we need to work with heterogeneous systems from multiple sources then the problems like poor design becomes prominent and significant. In this student record management system none of the database is properly designed and in some cases, there is no database at all. The databases are not normalized. Each of the campus maintains two "tables" to store student information. I have used double quotes as the word table is not used in its literal meaning, especially in the case of a single flat text file.

Student Table:
In each database Student table is used to maintain personal records of the students. This table has only one entry for each student in each campus. A student may have entries in student tables of two campuses in the issues like transfer cases.

Registration Table:
Second table is registration table that maintains the record for course registration. This table contains as many records for each student as many times he/she registered any course.

Each campus keeps two tables does not mean that each campus has two files only (one for each table). Each campus maintains its information independent of each other. Lahore campus maintains two text files for each batch i.e. entry taken in a year. For each batch one file contains student information and other file contains registration information. For eleven batches of BS Lahore campus has 22 text files. For four batches of MS Lahore campus contains eight text files. Same is the mechanism used in Peshawar campus to store the data in text files. Islamabad campus has MS Access d atabase with three tables. Two of these three tables contain student information. One table for MS and the other for BS students. The third table contains Registration data for both degree programs i.e. MS and BS. Karachi campus manages to store all this information in MS Excel sheets. Three Excel Books are maintained. Two out of three contains registration records (one for BS and the other for MS) and the third one contains student records for both degree programs.

Let us discuss "student record management systems" at each of the campuses.

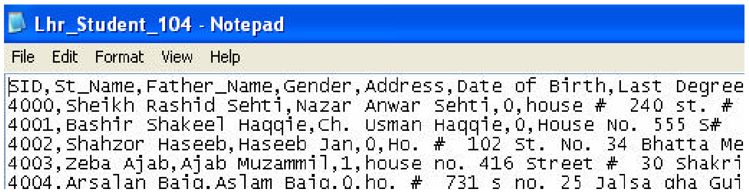
Data from Lahore Campus

- Data at Lahore campus is stored in Text files
- To store data regarding one complete batch 2 text files are used:
  - Lhr_Student_batch (Student record)
  - Lhr_Detail_batch (Course Reg. record)
- 22 text files for 11 BS batches
- 8 text files for 4 MS batches

The slide is self explanatory. Here batch is the year the student entry was taken i.e. 94, 95,…. 104 i.e. year 2004.

Data from Lahore Campus: Sample

- Flat file student data at Lahore campus



```
SID,St_Name,Father_Name,Gender,Address,Date of Birth,Last Degree
4000,Sheikh Rashid Sehti,Nazar Anwar Sehti,0,house #  240 st. #
4001,Bashir Shakeel Haqqie,Ch. Usman Haqqie,0,House No. 555 S#
4002,Shahzor Haseeb,Haseeb Jan,0,Ho. #  102 St. No. 34 Bhatta Me
4003,Zeba Ajab,Ajab Muzammil,1,house no. 416 Street #  30 Shakri
4004,Arsalan Baig,Aslam Baig,0,ho. #  731 s no. 25 Jalsa gha Gui
```

The slide shows the screenshot of a sample text file for student records at Lahore campus. We can see that the first row contains the header and the columns are delimited by comma. Let's discuss header of both student and registration tables in detail.

Lahore: Header of Student Table

- *SID:* Student ID
  - A numerical value, starting from 0
  - Starts from 0 individually for both degrees BS & MS
  - It is unique within a degree (BS/MS) but not unique across the degrees
  - Combination of SID and degree is always unique within a campus
- *St_Name:* Student name
- *Father_Name:* Father name

The slide is self explanatory.

Lahore: Header of Student Table

- *Gender:*
  - 0 for Male
  - 1 for Female
- *Address:* Permanent Address
- *[Date of Birth]:*
  - 14-Apr-1980
- *[Reg Date]:* Date on which student was enrolled

This is the convention used for storing some critical data at the Lahore campus. There is no guarantee that the same convention will be used at other campuses too, actually in some cases the converse may be true. We will identify and work on these apparent anomalies in the data profiling phase before we do the actual transformation.

Lahore: Header of Student Table

- [Reg Status]:
  - ‘A’ if student was enrolled as new Admission
  - ‘T’ if student was enrolled as Transfer case
- [Degree Status]:
  - ‘C’ (complete) if student has graduated
  - ‘I’ for incomplete degree
- *[Last Degree]:*
  - F.Sc. / A level for BS
  - M.Sc. / BS / BE for MS

The slide is self explanatory.

Lahore: Header of Course Reg. Table

- *SID:*
- *Degree:* BS/MS
- *Semester:* e.g. Fall04
- *Course:* Course code
- *Marks:* Out of 100
- *Discipline:* CS/TC/SE/CE

The slide shows the header and sample values for Course registration table at Lahore Campus.

Lahore: Facts About Data

- *Total students = 5,200*
- *Total male students= 3,466*
- *Total BS students= 4,400*
- *Number of graduated students= 3,200*
- *Number of post graduated std.= 600*

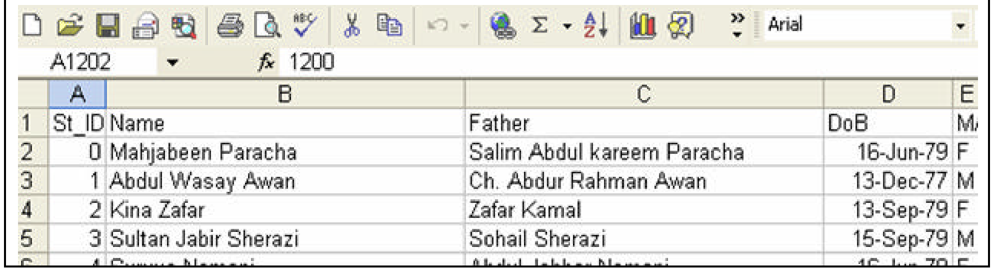
The slide shows some of the facts about Lahore campus. These facts can be used for data validation in later steps. However, this has to be taken with a "pinch of salt" because the facts before resolving the data quality issues will most likely be different as compared to the ones after the data has been cleansed.

Data from Karachi Campus

- Data at Karachi campus is stored in MS-Excel books
- Three books are maintained
  - STUDENT_KHR (Student record)
  - Reg_BS_KHR (BS course Reg. record)
  - Reg_MS_KHR (MS course Reg. record)
- STUDENT_KHR keeps two sheets
  - 'BS' for BS students records
  - 'MS' for MS students records

The slide is self explanatory.

Data from Karachi Campus: Sample



A1202 1200

	A	B	C	D	E
1	St_ID	Name	Father	DoB	M/
2	0	Mahjabeen Paracha	Salim Abdul kareem Paracha	16-Jun-79	F
3	1	Abdul Wasay Awan	Ch. Abdur Rahman Awan	13-Dec-77	M
4	2	Kina Zafar	Zafar Kamal	13-Sep-79	F
5	3	Sultan Jabir Sherazi	Sohail Sherazi	15-Sep-79	M

The slide shows MS Excel screenshot of the sample data for Karachi campus. Let's discuss its header in detail for both student and registration tables.

Karachi: Header of Student Table

- *St_ID:* Student identity
- *Name:* Student name
- *Father:* Father name
- *DoB:* Date of Birth
- *M/F:* Gender (M/F)
- *DoReg:* Date of Registration/Enrollment
- *RStatus:* Status of enrollment (A/T)
- *DStatus:* Status of Degree (C/I)
- *Address:* Permanent address
- *Qualification:* Last degree achieved

The slide is self explanatory.

Karachi: Header of Course Reg. Table

- *SID:*
- *Courses:* Course code
- *Score:* Out of 100
- *Sem:* e.g. Fall04
- *Disp:* CS/TC/SE/CE

The slide is self explanatory.

Karachi: Facts About Data

- *Total students = 6,000*
- *Total male students= 4,500*
- *Total BS students= 4,000*
- *Number of graduated students= 3,500*
- *Number of post graduated std.= 1,500*

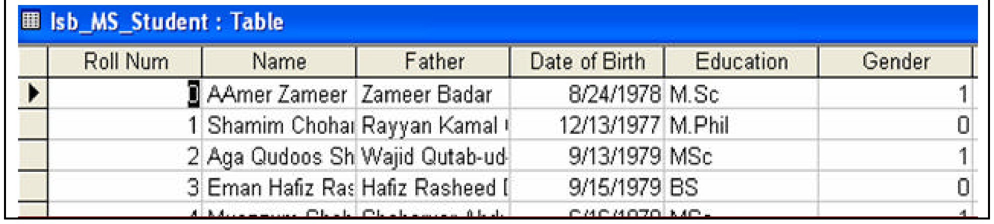
The slide shows some of the facts about Karachi campus. These facts can be used for data validation in later steps. Again we have to look at the facts keeping in mind that the same may change after data has been cleansed.

Data from Islamabad Campus

- MS-Access is used at Islamabad campus
- Database has three tables
  - Isb_BS_Student (MS Student record)
  - Isb_MS_Student (BS Student record)
  - Registration (All reg. record BS + MS)
- Roll number is also used as primary key in student table

The slide is self explanatory.

Data from Islamabad Campus: Sample



Isb_MS_Student : Table

Roll Num	Name	Father	Date of Birth	Education	Gender
0	AAmer Zameer	Zameer Badar	8/24/1978	M.Sc	1
1	Shamim Chohai	Rayyan Kamal	12/13/1977	M.Phil	0
2	Aga Qudoos Sh	Wajid Qutab-ud-	9/13/1979	MSc	1
3	Eman Hafiz Ras	Hafiz Rasheed [	9/15/1979	BS	0
4	[illegible]	[illegible]	[illegible]	[illegible]	1

The slide shows MS Access screenshot of the sample data for Islamabad campus. Let's discuss its header in detail for both student and registration tables.

Islamabad: Header of Student Table

- *Roll Num:* Student identity
- *Name:* Student name
- *Father:* Father name
- *Reg Date:* Date of Enrollment
- *Reg Status:* Status of Enrollment (A/T)
- *Degree Status:* Status of Degree (C/I)
- *Date of Birth:* Date of Birth
- *Education:* Last degree achieved
- *Gender:* Gender (Male=1, Female =0)
- *Address:* Permanent address

The slide is self explanatory.

Islamaba d: Header of Course Reg. Table

- *Roll Num:*
- *Course:* Course code
- *Marks:* Out of 100
- *Discipline:* CS/TC/SE/CE
- *Session:* e.g. Fall04

Here we can see that Degree (BS/MS) is missing, whereas same table contains records for both. Only way to differentiate is through discipline attribute.

Islamabad: Facts About Data

- *Total students = 4,400*
- *Total male students= 3,700*
- *Total BS students= 3,200*
- *Number of graduated students= 2,500*
- *Number of post graduated std.= 900*

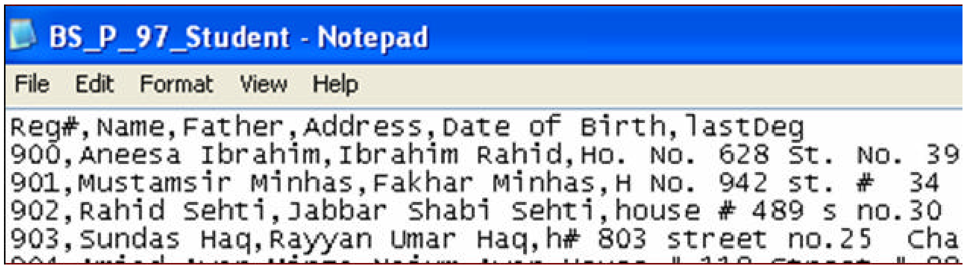
The slide shows some of the facts about Islamabad campus. These facts can be used for data validation in later steps.

Data from Peshawar Campus

- Data at Peshawar campus is stored in Text files
- To store data regarding one complete batch 2 text files are used
  - Lhr_Student_batch (Student record)
  - Lhr_Detail_batch (Course Reg. record)
- 22 text files for 11 BS batches
- 8 text files for 4 MS batches

The slide is self explanatory.

Data from Peshawar Campus: Sample



```
Reg#,Name,Father,Address,Date of Birth,lastDeg
900,Aneesa Ibrahim,Ibrahim Rahid,Ho. No. 628 St. No. 39
901,Mustamsir Minhas,Fakhar Minhas,H No. 942 st. #  34
902,Rahid Sehti,Jabbar Shabi Sehti,house # 489 s no.30
903,Sundas Haq,Rayyan Umar Haq,h# 803 street no.25  Cha
```

The slide shows the screenshot of a sample test file for student records at Peshawar campus. We can see that the first row contains the header and the columns are delimited by comma. Let's discuss header of both student and registration tables in detail.

Peshawar: Header of Student Table

- *Reg#:* Student identity
- *Name:* Student name
- *Father:* Father name
- *Address:* Permanent address
- *Date of Birth:* Date of Birth
- *lastDeg:* Last degree achieved
- *Reg Date:* Date of Enrollment
- *Reg Status:* Status of Enrollment (A/T)
- *Degree Status:* Status of Degree (C/I)

The slide is self explanatory.

Peshawar: Header of Course Reg. Table

- *Reg#:*
- *Courses:* Course code
- *Score:* Out of 100
- *Program:* CS/TC/SE/CE
- *Sem:* Fall/Spring
- *Year:* YYYY e.g. 1999

Here we need to identify semester session (fall04) through combination of Sem and Year

Lab Exercise

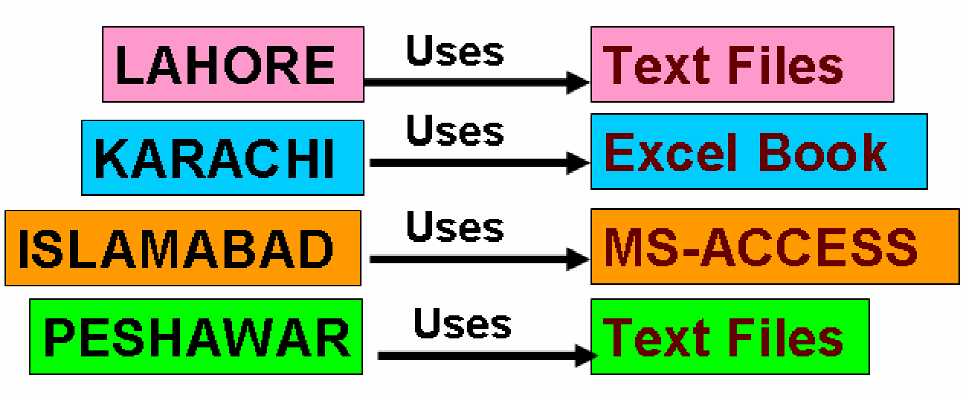
- Collect demographics for Peshawar campus
- Figure out problems in data at Peshawar campus
- Suggest suitable solutions to the problems identified above

Here is a small exercise. You are required to find the facts for the Peshawar campus. What problems are there in the data? And what, in your opinion, could be possible solutions for those problems.

Now by looking at each of the campus data individually, we found following problems that need to be considered and solved properly before extracting the data and ultimately loading it into the central repository.

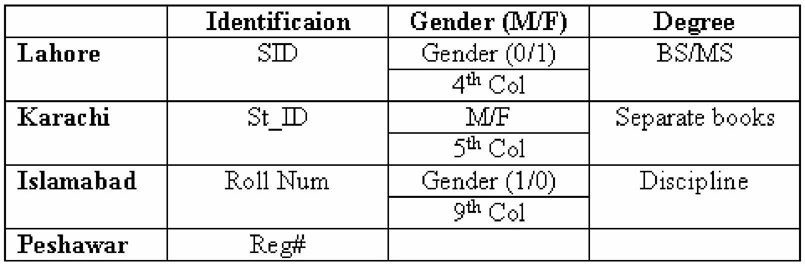
Problem-1: Non-Standard data sources

- Each campus uses data sources independent of other campuses



The major problem is the inconsistent data sources at different campuses. The slide summarizes the data sources at four campuses. We can see that Lahore and Peshawar campuses are using text files while Islamabad and Karachi campuses are using MS Access and MS Excel respectively.

Problem-2: Non-standard attributes



	Identificaion	Gender (M/F)	Degree
Lahore	SID	Gender (0/1)	BS/MS
		4th Col	
Karachi	St_ID	M/F	Separate books
		5th Col	
Islamabad	Roll Num	Gender (1/0)	Discipline
		9th Col	
Peshawar	Reg#		

The second problem is non standardized attributes across campuses. While looking at the header of data from different campuses we came to know the following problems regarding attributes and is summarized in the table in the slide.

Each of the campuses uses different attribute name for the identification or primary keys e.g. Lahore uses *SID* while Peshawar uses *Reg#* and so on.

Different conventions for representing Gender across the campuses e.g. Lahore campus uses 0/1 while Islamabad uses 1/0 for representing male and female respectively.

Similarly, there are different conventions for representing degree attribute across different campuses.

Problem-3: No Normalized database

- **None** of the campuses uses well designed normalized database
- Each campus uses two “tables”:
  - One table to store students’ personal data
  - Second table to store course registration data of each student
- Each campus uses multiple files to store these two tables

Actually Lahore, Karachi and Peshawar campus does not have databases at all, so there is no concept of normalization. These campuses maintain the data in sample shown as follows:

Lhr_detail_94 : Is a text file that contains the following details:

SID,Degree,Semester,Course,Marks,Discipline

Lhr_student_94: Is a text file that contains the following details:

SID,St_Name,Father_Name,Gender,Address,Date of Birth,Last Degree _

Note pad: Issues (1)

- Use of text files in record management systems is least suitable
- We cannot run any query on text file
- We cannot validate any input to text file
- Comma is used as a field separator, any erroneous placement of comma can spoil the whole record
- There is no technical way of locating any particular record

Having discussed the three major problems, lets now look at what are the issues regarding the record management tools at individual campuses. This slide and the following four list the issues related to Notepad.

Note pad: Issues (2)

- If I want to locate the record of 'Mohammad Ali Nawaz' and I do not know his roll number, what would I do?
- At Lahore campus, academic officer used to do it by "Find" option of text file
- Is it a proper way? Does it work always?
  - What about 'Mohamed Ali Nawaz'?

People at different campuses, including the Lahore campus have developed ways and means to answer some questions. But these so called "techniques" have their own inherent limitations. For example, if I want to find the information about a student named 'Mohammad Ali Nawaz' I can use the find command from the notepad, but what if there is a slight change in the spelling? Of course the technique is not going to work.

Note pad: Issues (3)

- If I want to count total students who belong to Multan, can I do it in note pad?
  - No
- To achieve this purpose, admin at Lahore used to open the file in Microsoft Word. Then use "Replace with" functionality of Microsoft Word to count total occurrences of Multan.

In 'Replace With' dialog box if I enter 'Multan' is replaced with 'Multan' & use 'Replace All' option. I can get the total occurrences of Multan. *Interesting*

Some simple questions that can be answered if there was a database can not be answered, such as the number of students from any particular city. There can be number of short-term self-developed ad-hoc mechanisms, but they are not guaranteed to succeed and have their own inherent limitations.

Note pad: Issues (4)

- Some improper ways can work for very limited cases
- We can't collect demographics in note pad
  - Total number of male students
  - Students with a particular age
  - Students with a particular educational background
  - Students with a particular CGPA
  - Etc.

Some very simple statistics can not be collected in the absence of a database as we have a big text file. Some of the examples are number of male students or students of particular age. We can get answer to these problems by parsing the files, but text parsing is not only very slow, but is also very complicated. All these complications and inefficiencies can be reduced, and even removed if he had a database in place.

MS-Excel: Issues (1)

- Karachi campus uses MS-Excel sheets to maintain students record
- MS-Excel is again not basically developed for this purpose
- However, it works somewhat better than note pad, as it can answer to more questions but once again in an improper way
- Both methods adopted for notepad are available here also but it can work more than that

MS Excel is better than having a big text file. For example Excel supports some simple tests and other commands that can help more efficiently answer the questions that could not be answered using a plain text file. But, still Excel is not the right way to store and keep the data for a host of reasons, that we discussed in Lect-6 of the theory part.

MS-Excel : Issues (2)

- Now, I can count total number of male or female students?
- I can sort all columns on basis of gender and get all males and females clustered
- I can get student-wise particular scores
- I can get answers to many questions through conditional queries supported by MS-Excel

The slides gives a way of finding answer to some questions, but remember that we are dealing with large data sets, and for such large sets comparison sort which at best is O(n log n) really hurts.

MS-Excel : Issues (3)

- Maintenance of records in MS-Excel is better with respect to the data quality concerning issues
- MS-Excel recognizes the correct data type of columns
- It somewhat validates the input, i.e. illegal input is filtered

Some more benefits of Excel. At least there is a column type i.e. not all values are textual in nature, and this helps in the context of data validation.

MS -Access: Issues (1)

- MS-Access is a proper RDBMS and can work well for small databases
- At Islamabad campus, the problem is the poor design of database, not the tool
- SQL of MS-Access is not very powerful, like that of SQL Server, but it works fine to maintain records at campus level

Finally Islamabad campus at least is using the right tool i.e. Access databases, but it works for small personal databases not years of data of a single campus and then pooling together the data of multiple campuses. Thus the problem is not of the poor design (As there is no real design) but of the wrong tool. The correct choice could have been to use MS SQL server which can handle larger work loads more gracefully.

Problem Statement

- We have disparate sources of data
- We have to implement single source of truth i.e. DWH, so that decision makers can be supported to get detailed or summarized university level view, irrespective of particular campus
- In the lab exercises and working we will experience interesting and complicated issues need to be handled while moving towards single standardized source.

Thus, in view of the issues and challenges in our simulated scenario of a multi-campus university, the problem ahead can be summarized as under.

There are disparate and diverse data sources and we have to implement a DWH i.e. single source of truth that can support the decision making at the head office.

Lab Lec: 3
Extracting Data Using Wizard

Steps towards single source of truth

- Identify source systems
- Figure out the issues associated with each source system
- Extract data
- Transform data
- Load data
- Quality checks

Data warehouse is a single source of truth. We have to put all departmental data from desperate sources at one place in some standard form. The task is not trivial. Desperate sources of data have a lot of inherent issues. High level steps given in the slide gives just an overview of the task. First of all we have to identify the source systems. It is quite possible that each department uses different database systems or same organization at different geographical locations uses different database management systems. To put data into a single source after extracting from such diverse sources requires powerful tools especially designed to fulfill the purpose. We will use Microsoft SQL Server which is a user friendly graphical tool and makes such a complex task doable by some practice.

After identification of source systems, it is necessary to study the issues that must be considered before putting all the data together. Microsoft SQL Server provides a powerful support to perform Extract, Transform and Load (ETL) data from source systems to Destination system. Finally certain steps are performed to check and improve quality of d ata.

In this lab exercise we will perform all the above steps in a detailed manner through powerful support of Microsoft SQL Server.

Example: Student Record System: Diversity

- Identify source systems at different campuses

- Source systems are as follows
    - Lahore campus uses simple Text Files
    - Karachi campus uses MS-Excel workbooks
    - Islamabad campus uses MS-Access DB
    - Peshawar campus uses simple Text Files

The task starts from analysis of source systems at different campuses. Data for Lahore and Peshawar campus is kept in simple text files, for Karachi Ms-Excel books are used and for Islamabad MS-Access is used to keep data. Further more table structures, date formats, conventions for gender M/F or 1/0, etc. are different from campus to campus.

First we will load the data for each campus in MS-SQL Server as it is, in different databases then before putting all the pieces together all these issues will be addressed.

Example: Student Record System: Issues

- Figure out the issues related with each source system
- Issues include
  - Standards and formats of stored data
  - Number of tables in different source systems
  - Type of columns, their number and ordering in different tables to be combined

Here we need to figure out the issues in source systems. As source data is distributed over different campuses therefore the issues like difference in date formats, conventions of storing gender (M/F,0/1,1/0), etc are obvious. Microsoft SQL Server has a good support to resolve these issues.

Extracting University Data

1. Lets start our practical by loading data for the university
2. We have data from four different campuses
3. Initially we will develop four different databases, one for each campus, and load corresponding data
4. Then we will transform and standardized each database
5. Finally we will combine all the four databases to single source of truth (DWH)
6. At each step we will run queries to collect demographics

For loading data for the university, it is required to load the data for four campuses, separately and as it is, into the MS-SQL Server. Once all data is loaded to SQL Server then the tasks of transformation and standardization would be started. First we will transform the database of each of the campuses individually. Then we will standardize the databases of four campuses separately. Finally, the data from four different campuses will be put together.

Extracting Data Using Wizards

- *Import and Export Data Wizard* provides the easiest method of loading data.
- The wizard creates package which is a collection of tasks
- Tasks can be as follows:
  - Establish connection through source / destination systems
  - Creates similar table in SQL Server
  - Extracts data from text files
  - Apply very limited basic transformations if required
  - Loads data into SQL Server table

After addressing the issues we decide to select a suitable tool in SQL Server to resolve these issues. At this stage we are not performing transformations rather we are just copying data from source to destination. For this purpose the easiest method is the use of wizard. Wizard would create package for us including all required tasks as:

- Establishes connection through source / destination systems

- Creates similar table in SQL Server
- Extracts data from text files
- Applies very limited basic transformations, if required
- Loads data into SQL Server table

Extracting Data for Lahore Campus

- **First of all load data for the Lahore campus**

1. Connect to source Text files
2. Connect to Destination SQL Server
3. Create new database 'Lahore_Campus'
4. Create two tables Student & Registration
5. Load data from the text files containing student information into Student table
6. Load data from the text files containing registration records into Registration table

- **Import/Export Wizard is sufficient to perform all above mentioned tasks easily**

Loading data for Lahore campus includes following tasks:

1. Connect to source Text files

Since there are many text files for Lahore campus, we need to load those text files separately. First of all, select the file that is to be loaded first.

2. Connect to Destination SQL Server

In this case our source system is a text file. For transformation and standardization we will load all data as it is from source file to the SQL server and then through powerful tools of SQL Server, we will perform these intended task.

3. Create new database 'Lahore_Campus'

To load data for four campuses we will develop four separate databases. So, to load data for Lahore campus we will create a new data base named 'Lahore_Campus'.

4. Create two tables Student & Registration

All files containing student information will be loaded in one table Student and all other files containing registration information will be loaded in other table Registration. After this step we will have two populated tables only.

5. Load data from the text files containing student information into Student table.

6. Load data from the text files containing registration records into Registration table

Import/Export Wizard is sufficient to perform all above mentioned tasks easily. So we will use the wizard as it can provide us good functionality in this scenario.

Seven Steps to Extract Data Using Wizard

1. Launch the Wizard
2. Choose a Data Source
3. Choose a Database
   - Specification of file format incase of Text files
4. Specify the Destination
5. Choose Destination Database
   - Selection of existing database or creation of a new database
6. Select a table
   - Selection of existing table or creation of a new table
7. Finalizing and Scheduling the package

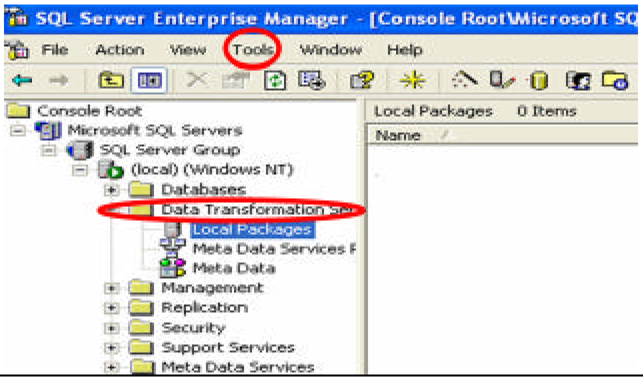
The slide states seven simple steps to create a package for data loading through wizard. Lets discuss each o f the steps in detail.

Step1: Launch the wizard(1)

- **Two methods to launch the wizard**
  - Start > Programs > Microsoft SQL Server > Import & Export Data
  - Start > Programs > Microsoft SQL Server > Enterprise Manager
    1) On console root drop Data Transformation Service node
    2) Tools > Data Transformation Service > Import/Export data

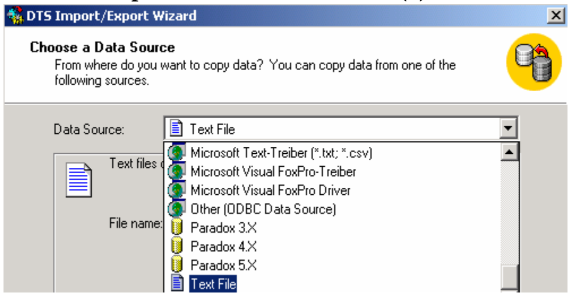
These are two different methods to launch the wizard. We can use either.

Step1: Launch the wizard(2)



The slide shows the main screen of SQL Server enterprise manager. In the left pane we have Console root. We can see Data Transformation services highlighted. Expand the node mentioning Data Transformation Services and then press Tools in the menu bar. This will lead you to launch the wizard to load data.

Step2: Choose a Data Source(1)


DTS Import/Export Wizard
Choose a Data Source
From where do you want to copy data? You can copy data from one of the following sources.
Data Source:
Text File
Microsoft Text-Treiber (*.txt; *.csv)
Microsoft Visual FoxPro-Treiber
Microsoft Visual FoxPro Driver
Other (ODBC Data Source)
Paradox 3.X
Paradox 4.X
Paradox 5.X
Text File
Text files
File name:

After launching the wizard, first of all we will be welcomed by the wizard through a welcome screen having a button 'NEXT'. On pressing 'NEXT' we will see the above window. This window is basically allows to perform step 2 of creating a package through wizard. In this window we can see a drop down menu 'Data Source'. This menu shows all possible connections that are available through SQL Server. Connection will be selected according to sou rce system. In our case, source system is a simple text file. Therefore select the last option text file from this drop down menu.

Step2: Choose a Data Source(2)

- Data source for Lahore is a Text file
- After specifying data source, Browse first file to be loaded from Lahore data
- "Lhr_Student_94.txt", is a text file that contains students data of Lahore campus
- First of all browse this file to load

After selecting text file as a source system, we will see a new option on the lower part of same window. The option is to browse the text file considered as a source system. To load data for Lahore campus first of all select the file **'Lhr_Student_94.txt'**. Browse option allows us to have an Open File dialog box, which will let us to locate file through navigation. We will locate the directory in which our source data is placed, select the file 'Lhr_Student-94' from Lahore folder. Then press next to move to step 3 of wizard.

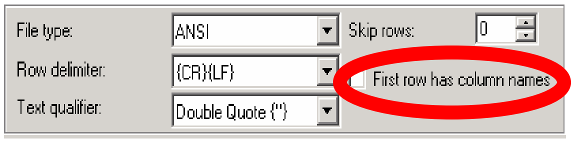
Step3: Specify File Format(1)

- **After specifying file, Wizard asks for file format like**
  - Columns are fixed length fields or delimited by some character
  - First row contains header information (column/field) names or directly data
  - What are the text qualifiers
  - What is the file storage format
  - How many rows, if required, should be skipped from start

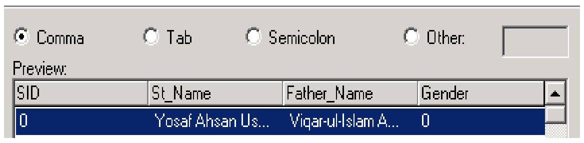
Third step in creation of package through wizard is to specify the format of the file to be loaded. In this new screen we are supposed to provide following information to the SQL Server:

- Whether the source file has fixed length columns or delimited length columns.
- Does the first row contain column name?
- Do we have text qualifiers in our source text file? If yes then select the specified from drop down list menu.
- What is the file storage format?
- Do we want to skip rows from start of file? If yes, then how many rows are supposed to be skipped?
- The answers to all above stated questions are provided through graphical user interface objects like check box, radio buttons and drop down list menus.

Step3: Specify File Format(2)



File type: ANSI
Skip rows: 0
Row delimiter: {CR}{LF}
First row has column names
Text qualifier: Double Quote (")



[✓ Comma] Tab Semicolon Other:

Preview:

SID	St_Name	Father_Name	Gender
0	Yosaf Ahsan Us...	Viqar-ul-Islam A...	0

Formatted: Font: 7 pt

Here we can see the screen to input answers of the questions asked on previous slide. Each and every object is self explanatory. First of all you will see the above screen and then on pressing next we need to specify the column delimiter that is used in our text file.

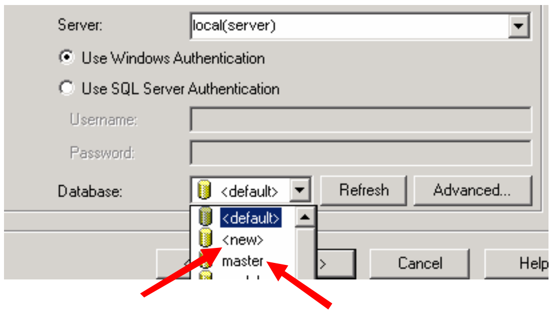
Step4: Specify the Destination(1)

- As many options for destinations are available as were for the source (in step 2)
- By selecting Text files as destination, data extracted from text file would be stored in another text file
- Incase of given scenario we want to load data in SQL Server, which is a default option for destination

Step 4 in creation of a package through wizard is to specify the destination database. This needs to establish a connection with destination system first. For this purpose select the correct connection from the drop down menu **'Destination'** which contains all possible connection options as were available for source in step 2.

In case we select Text file as a destination then the data extracted from one text file will be loaded in another text file. As we have planned to load data in SQL Server, therefore select SQL server connection, which is by default selected.

Step4: Specify the Destination
Choose Database

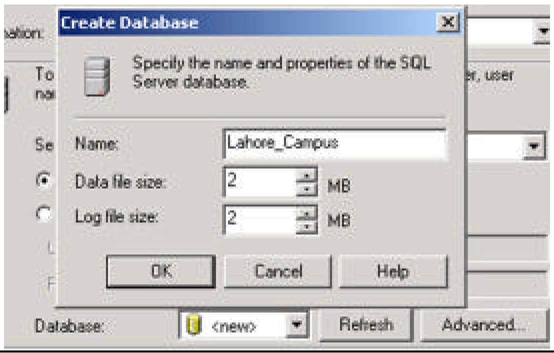


Formatted: Font: 7 pt

Formatted: Font: 7 pt

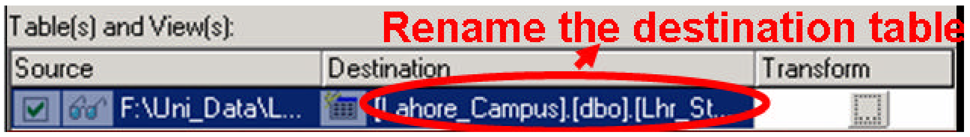
On selecting SQL Server as destination we can see options appeared on the same window. From here we are just required to select the destination database in destination system. This can be done through a drop down list menu titled as Database. This drop down list menu is showing all databases available to this user in SQL server. **<new>** option allows us to create a new database. We will select new as we want to create separate database initially for each campus.

Step5a: Creating a New Database



On selecting **<new>** option we can see this Create database pop up dialog box. This dialog box is asking for the name of database and the space needed for that database. Write name as ‘Lahore_Campus’ and then press OK. Then press NEXT

Step6: Creating a New Table



Table(s) and View(s):

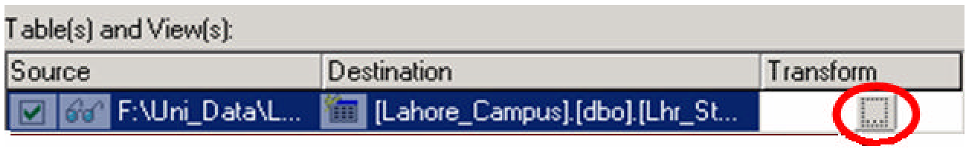
Source	Destination	Transform
F:\Uni_Data\L...	[Lahore_Campus].[dbo].[Lhr_St...	

- Source contains the name of input file
- Destination is a new table with same structure as that of source text file
- By default, name of destination table is same as that of text file i.e. "Lhr_Student_94", Rename it as "Students"

After selecting the destination database from existing databases or new database, we press next. This time we will see the above window on the screen. This window contains 3 columns source, destination and transform. Source contains the name of input file. Destination shows the name of the destination table in destination database which was selected in previous step. The wizard checks the names of all available tables in destination database. Select the table for loading whose name is exactly the same as of source table/ view/ text file. If none of the table has matching name then wizard generates code to create new table with exactly the same name and same number of columns as that of source table/view/text file. Wizard generates table with all columns having same data type i.e. VARCHAR(255)

In this case our destination database is empty, as it has not yet physically created. Therefore, wizard selects the name for new table 'Lhr_Student_94' as it is the name of source text file. To rename this text file we can double click the destination highlighted row. As shown in the figure.

Step6: Creating a New Table: Transform



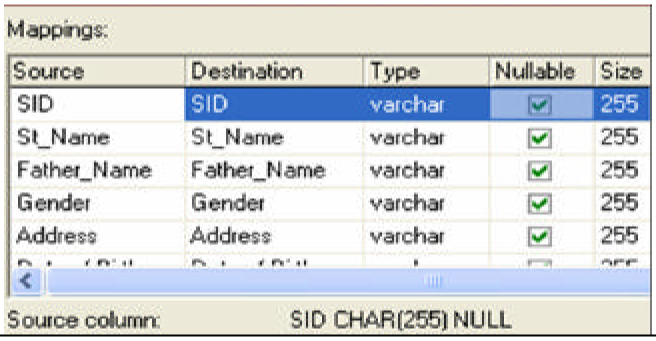
Table(s) and View(s):

Source	Destination	Transform
F:\Uni_Data\L...	[Lahore_Campus].[dbo].[Lhr_St...	

- Name and order of columns in new table can be seen through "Transform"
- Names, data types and order of columns in new table can be changed through "Transform"

Transform column is used to apply transformations that are available through wizard like changing the data types of the columns, order of the columns and so on. If we click the transform button, we can see the mapping between source and destination columns.

Step6: Creating a New Table
Column Mappings


Mappings:

Source	Destination	Type	Nullable	Size
SID	SID	varchar	✓	255
St_Name	St_Name	varchar	✓	255
Father_Name	Father_Name	varchar	✓	255
Gender	Gender	varchar	✓	255
Address	Address	varchar	✓	255

Source column: SID CHAR(255) NULL

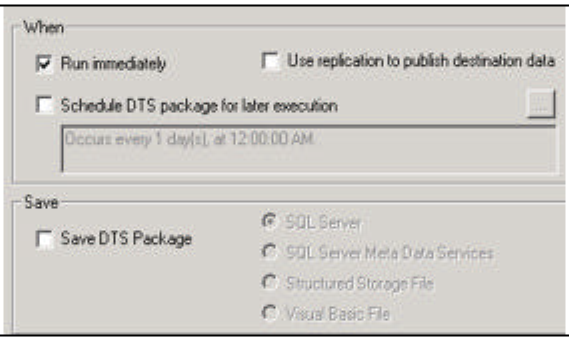
On clicking transform we can view the dialog box showing mapping between source and destination columns. The third column is showing data type of destination column. By default all types are selected as varchar of size 255. By default *Nullable* option is checked that means the corresponding columns can contain null values as well. A pointer variable varchar to hold character array of length 255 requires a lot of memory to be consumed especially when we are dealing with a huge input data. So we should change the type according to required lengths, for example, for Gender one character is enough as gender may be M/F or 0/1.

Step7: Scheduling the Package(1)

- In the preceding six steps, a package has developed including tasks of establishing connections, extraction and loading of data
- Whenever the package would run all the tasks would be executed

After finalizing the mapping we can press next for finalizing the package. On saving the package all steps would be written in a script file. Whenever the script would be run all steps would be executed and tasks would be performed.

Step7: Scheduling the Package(2)


When

☑ Run immediately ☐ Use replication to publish destination data

☐ Schedule DTS package for later execution

Occurs every 1 day(s), at 12:00:00 AM

Save

☐ Save DTS Package

● SQL Server

○ SQL Server Meta Data Services

○ Structured Storage File

○ Visual Basic File

The above dialog box provides following options:

1. Run the package immediately
2. Schedule DTS package to run periodically
3. Use of replication to publish destination data (we are not concerned with this option currently)
4. Save package

For saving package we need to select the destination location that can be one of following:

a. SQL Server
b. SQL Server Meta Data Services
c. Structured storage file
d. Visual Basic file

Now, we have successfully completed the package and it is ready to run. It can be executed manually or automatically by scheduled option.

Execution of a package

1. Connection with source (Text file) is established
2. Connection with destination (MS-SQL Server) is established
3. New Database at destination is created
4. New table is created
5. Data is extracted from source
6. Data is loaded to destination

When this package will be executed either automatically due to scheduled option enabled or manually following tasks will be performed:

- Connection with source (Text file) is established
- Connection with destination (Ms-SQL Server) is established
- New Database at destination is created
- New table is created
- Data is extracted from source
- Data is loaded to destination

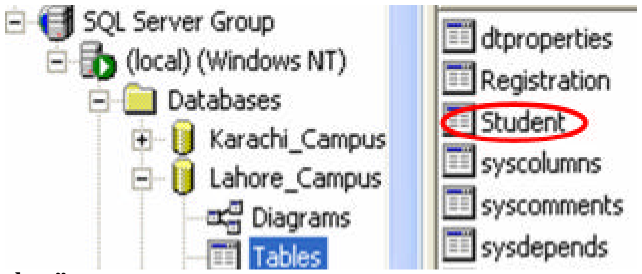
Execution can be completed successfully or it may be stopped and roll backed due to some error. In case of successful completion of execution all the transactions will be committed to the database otherwise, if some error occurs, execution will be terminated abnormally and all transactions will be rolled back. In second case when we will access the database we will find it in the state that was before the execution of package.

Verification of Results(1)

- Results can be verified by view resultant table and its rows
- New Database "Lahore_Campus" can be accessed through SQL Server Enterprise Manager
- Expand the tree on the left pane
- Local > Databases > Lahore_Campus > Student

After successful completion of the package we can verify the results by viewing the destination table and its rows. To access new database 'Lahore_Campus", we will use SQL Server Enterprise Manager. In SQL Server Enterprise Manager we will expand the console tree in the left pane and drop the databases, there we can find 'Lahore_Campus'. Double click the node tables and locate Student table in right pane window.

Verification of Results(2)



Right click "Student"
Open Table > Return all rows

Right click student table and select option '**Return all rows'** in sub menu open Table. This will show you all the rows loaded in the destination table.

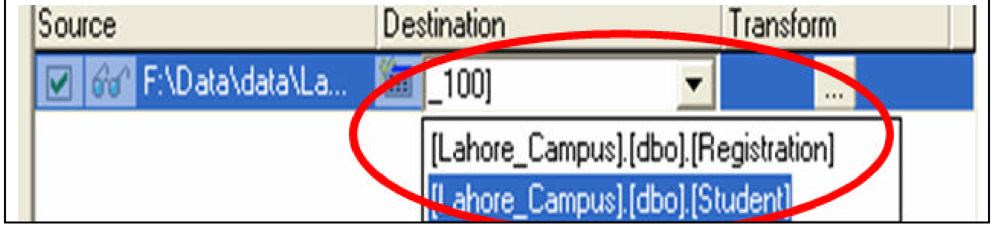
Adding More Records to Table

- The Loaded file contains the data for batch 1994 only
- Data for remaining ten batches is stored in ten separate text files
- In SQL Server we will be having only one student table for all batches in a campus, so data for remaining table is required to be added in same table

By this step destination table contains the data for one batch only, as the source file was for the 1994 only. Data for remaining ten batches is stored in separate text files. We are required to add the data for remaining batches in the same table. Following slides guide you for loading data in the same table.

Adding More Records to Table

- Repeat first five steps as it was done while loading previous table
- In step six, Drop down destination menu and select the table in which you want to append the records


Source
Destination
Transform
F:\Data\data\La...
_100]
[Lahore_Campus].[dbo].[Registration]
[Lahore_Campus].[dbo].[Student]

To load data in the same table, we are required to repeat first five steps as it was done before.

1. Launch the Wizard
2. Choose a Data Source
3. Choose a Database

Specification of file format incase of Text files

1. Specify the Destination
2. Choose Destination Database

Selection of existing database "Lahore_Campus"

The difference comes in 6th step when we select destination table. To choose destination table drop the menu in Destination column and locate the name of required table that is "Student". That is all we are required to add records in the same table.

Load All BS &MS Student Records For Lahore

- Repeat the same procedure for remaining nine files for the batches 1996 to 2004
- Then load "Course Registration Details" data in a new table "Registration"
- We have eleven files for Registration data, one for each batch
- Load all files in table, "Registration"

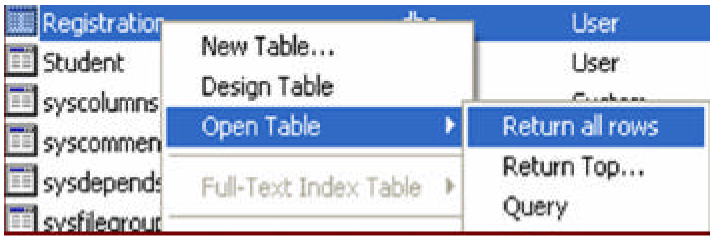
We are required to repeat the same procedure for remaining nine files for the batches 1996 to 2004. After loading all text files containing student data of Lahore campus, load "Course Registration Details" data in a new table "Registration". We can find eleven files for registration data, one for each batch. Load all course registration details files in table, "Registration"

Loading "Course Registration" Records

- Load "Course Registration" records, in the same way
- We have 11 BS + 4 MS text files for Registration data, one for each batch
- Load all files in table, "Registation"
- Registration records already contain a column indicating degree (BS/MS)

Similarly load all text files that contains course registration details records. We have 11 text files for BS registration detail records and 4 text files for MS registration detail records. Create new table registration and load all files in new table. After loading all course registration records for BS and MS we can see that each record would still be identified uniquely as registration table contains a column indicating t he degree of a student either BS or MS.

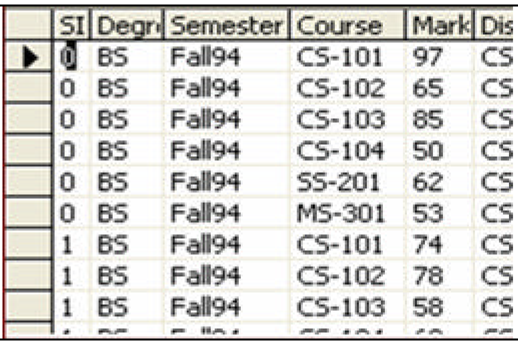
Verification: "Course Registration" Records



- Right click Registration table through enterprise manager, to get the above menu

In the same way as we did for Student table we may verify the results of package targeting Registration records load.

Verification: "Course Registration" Records

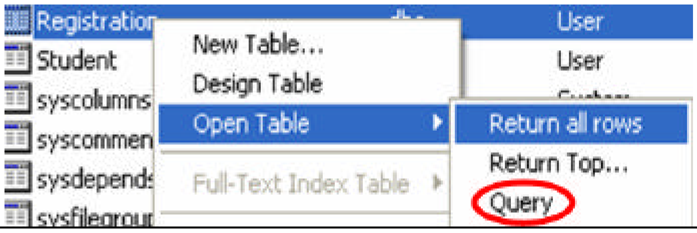


SI	Degr	Semester	Course	Mark	Dis
0	BS	Fall94	CS-101	97	CS
0	BS	Fall94	CS-102	65	CS
0	BS	Fall94	CS-103	85	CS
0	BS	Fall94	CS-104	50	CS
0	BS	Fall94	SS-201	62	CS
0	BS	Fall94	MS-301	53	CS
1	BS	Fall94	CS-101	74	CS
1	BS	Fall94	CS-102	78	CS
1	BS	Fall94	CS-103	58	CS

The above slide shows the output of query "**Return all rows**" from **Registration** table.

Demographics

- Finally all the data for Lahore Campus has been loaded
- Now we can collect demographics on this data through queries



By this time we have two tables in SQL Server, Students & Registration that contain all students and registration records for Lahore Campus respectively. Before starting transformation and quality check of data, it is required to collect demographics so that we can chalk out the way of transformation.

Total students & BS Students

- **Total Number of Students**
  Select COUNT(*) AS Expr1
  From Student
- **5,200**

- **Total BS students**
  ```
  SELECT     COUNT(*) AS Expr1
  FROM        Student
  WHERE     ([Last Degree] IN ('F.Sc.', 'FSc', 'A level', 'A -Level', 'HSSC'))
  ```
  - 4,400

Total number of students
This is the simplest query that counts the total number of students registered in Islamabad campus. For each campus individually, this query provides correct results but for consolidated data for all campuses this type of query will count those students twice who admitted in one campus and then transferred to another campuses as their names would be present in the databases of both campuses. While running queries for consolidated data we must take care of such issues.

Total BS students
Repeating IDs for BS and MS students make such queries complicated which involves the separation of BS and MS students. To count total number of BS students we must separate them from MS students. In **Student** table we do not have any such direct information that can filter BS students from total students. To meet such requirements we devised a methodology of considering degree information. No doubt this is not a perfect way of filtering as the quality of result can suffer a lot due to the presence of outliers. However, in real life when we have to deal with legacy systems we need to face hundreds of such complicated issues due to bad designs and limitations of legacy systems. In such scenarios we need to devise solutions intelligently through indirect ways.

SIDs vs. Students

- **Total unique SIDs**
  ```
  SELECT     COUNT(DISTINCT SID) AS Expr1
  FROM        Student
  ```
- **4,400**

- **SIDs are exactly equal to the BS students**

Total unique SIDs
In this query we counted the distinct student IDs from student table. The answer is exactly equal to the total number of BS students because the student IDs are unique among BS students only and same IDs are repeating for BS students.

Unique identification of each student
Unique identification of each student is possible through combination of degree only. In one campus there can be only one such student who has SID=1 and he is enrolled in BS degree program. Similarly there can be only one student is possible who has again SID=1 and is enrolled for MS degree program. Each SID can be repeated at most twice, one time for BS student and other time for MS student.

Repeating Student IDs for BS and MS

- Students IDs are repeating for MS students
- After loading records of MS students in the same table with BS students, student ID is no more useful to identify each record uniquely
- Now we need some more information to be used with IDs to identify each record uniquely

After loading all files for BS and MS, you will find an interesting problem. As the university was managing the records of BS and MS separately, Student ID was used as Primary key to identify students uniquely. Student ID is just an auto-increment sort of number which starts from zero for both BS and MS students. In warehouse environment when we have combined both BS and MS students, Student ID no more uniquely identifies each student in warehouse.

Solution of Repeating IDs

- Problem of repeating IDs can be resolved through the use of 'Last Degree' column with ID
- SID + [Last Degree] -> unique record
- 1) SID = '100', [Last Degree] = 'F.Sc.' 2) SID = '100', [Last Degree] = 'M.Sc.'
- 1) is a BS student & 2) is an MS student
- What can be outliers here?

As ID is no more useful we need to add some additional information with ID to identify each student uniquely. If we look at table structure of student table, we can find a column Last degree that can be used to distinguish BS students from that of MS students. But there is also a little chance of outliers like the students who are doing MS in computer science after MS in physics etc. These outliers, if exists, will be handled separately.

Solution of Repeating IDs *(Cont.)*

- Outlier can be a BS student who has already sixteen years education background (say) in Mathematics and he is again registered for BS computer science
- Such a student can have [last degree] = 'M.Sc.'

At this time we will use information like if student ID is 1 and last degree is 'F.Sc.' he is an undergraduate student, and if student ID is 1 and last degree is BS then he is a graduate student who is enrolled in MS.

Male Students

- **Total Number of Male Students**

```
SELECT     COUNT(*) AS Expr1
FROM        Student
WHERE     (Gender = '0')
```

- **3,466**

To find out the total number of male students we counted all records where gender = ‘0’. This gives 3,466. We can not say any thing about the quality of this result. The quality can be discussed after data profiling but not at this stage. There may be a lot of errors in data, we may have some other male students for whom gender is missing here. We may have some records with noise like ’01’ or ’10’ which out of domain of the column. But all such issues can be identified first in data profiling only not before that.

Female BS students in Telecom

- **Total Number of Female students in BS Telecom**

```
SELECT COUNT(DISTINCT r.SID) AS Expr1
FROM    Registration r INNER JOIN
           Student s ON r.SID = s.SID AND
            s.[Last Degree] IN ('F.Sc.', 'FSc',
            'HSSC', 'A -Level', 'A level') AND
             r.Discipline = 'TC'  AND s.Gender = '1‘
```

- **365**

This query requires access of both student and registration table. Gender of student can be found from student table where as Discipline can be found from registration table only. To find the answer of such a query we need to have inner join of both tables.

Extracting Data for Karachi Campus

- **Now load data for Karachi Campus**

1. Connect to source MS-Excel
2. Connect to Destination SQL Server
3. Create new database ‘Karachi_Campus’
4. Create two tables Student & Registration
5. Load data from the Excel worksheet containing student information into Student table
6. Load data from the Excel worksheets containing registration records into Registration table

- **Import/Export Wizard is sufficient to perform all above mentioned tasks easily**

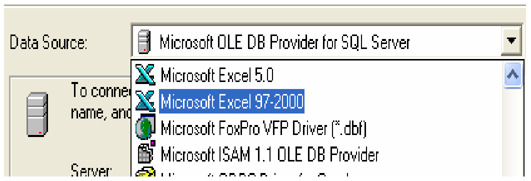
By this time we have loaded data and collected demographics for Lahore campus only. Now we are required to load data fro Karachi campus. For Karachi campus we need to load data from Excel files. Main steps of loading are as follows:

1. Connect to source MS-Excel
2. Connect to Destination SQL Server
3. Create new database ‘Karachi_Campus’
4. Create two tables Student & Registration
5. Load data from the Excel worksheet containing student information into Student table
6. Load data from the Excel worksheets containing registration records into Registration table

Again Import/Export Wizard is sufficient to perform all above mentioned tasks easily

Student data for Karachi

- Step1: Launch the wizard
- Step2a : Choose a data source



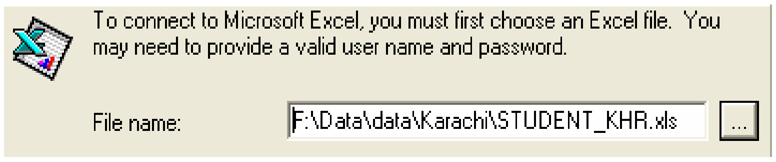
Data Source: Microsoft OLE DB Provider for SQL Server

Microsoft Excel 5.0
Microsoft Excel 97-2000
Microsoft FoxPro VFP Driver (*.dbf)
Microsoft ISAM 1.1 OLE DB Provider
To conne
name, and
Server:

Step 1 is same as it was for the text files. In step two we need to select Microsoft Excel 97-2000 as data source.

Student Data for Karachi *(Cont.)*

- Step2b: Browse Student Excel-Worksheet



To connect to Microsoft Excel, you must first choose an Excel file. You may need to provide a valid user name and password.

File name: F:\Data\data\Karachi\STUDENT_KHR.xls

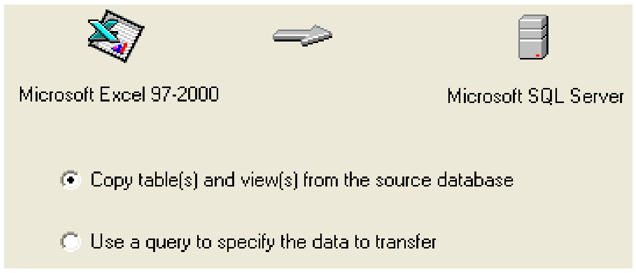
- STUDENT.xls for all BS/MS student records
- Reg_BS_KHR.xls for BS Registration records
- Reg_MS_KHR.xls for MS Registration records

After selection of MS-Excel data source, we are required to locate the data file that contains extension ".xls". To load student data we need to locate STUDENT.xls data file as it contains data for both BS and MS students.

When we will load registration data we will require to load Reg_BS_KHR.xls file for BS students registration details and Reg_MS_KHR.xls for MS students registration records.

Student Data for Karachi *(Cont.)*

- **Step 3:** Specify Destination
- **Step 4:** Choose/Create Database
  - Karachi_Campus
- **Step 5 a:** Table/View or query



(●) Copy table(s) and view(s) from the source database

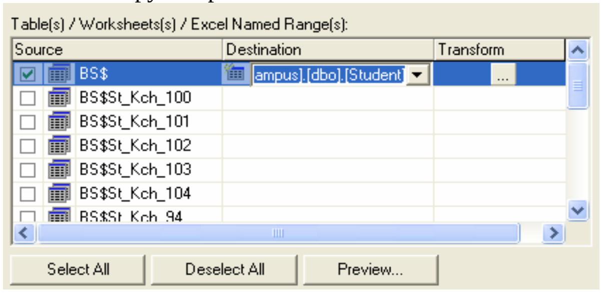
( ) Use a query to specify the data to transfer

In step3 we select Microsoft SQL Server as destination and in step4 we create new database Karachi_Campus in the same way that we adopted for Lahore_Campus.

As some particular sort of queries can be run in MS Excel therefore wizard asks us whether we want to extract data from source through query or we want to copy complete table/view. As we do not want to filter data through query therefore we will select the other option that is copy complete table/view.

Student Data for Karachi ***(Cont.)***

- **Step 5b:**
  - Choose BS to copy complete worksheet of BS students


Table(s) / Worksheets(s) / Excel Named Range(s):

Source	Destination	Transform
✓ BS$	ampus].[dbo].[Student]	...
BS$St_Kch_100		
BS$St_Kch_101		
BS$St_Kch_102		
BS$St_Kch_103		
BS$St_Kch_104		
BS$St_Kch_94		

In this screen we can see a lot of worksheets, where as in our Student_KHR.xls file there are only two worksheets, one for BS and one for MS. You can see in the dialog box that the name of worksheet BS is followed by '$'. To load complete worksheet BS select the first option 'BS$'. Following options show logical divisions with in a worksheet. Like in this case we can see an option 'BS$St_Kch_94', it means that by selecting this option only those students will be copied who belong to the batch 1994. Similarly, by selecting the option 'BS$St_Kch_100' only students of batch 2000 will be copied but if we select the option BS$ then all students in worksheet BS irrespective of their batch will be copied.

Student Data for Karachi ***(Cont.)***

- **Step 5b** ***(Cont.)*** **:**
  - In step 5b you can see two columns Source and Destination
  - If we want to copy all records from BS worksheet we need to check 'BS$'
  - If we want to copy only those records from BS worksheet that belongs to year 2000 we need to choose BS$St_Kch_100
  - Similarly BS$St_Kch_101 belongs to records of year 2001 and so on

In step 5b we can see two columns Source and Destination. If we want to copy all records from BS worksheet we need to check 'BS$'. If we want to copy only those records from BS worksheet that belongs to year 2000 we need to choose BS$St_Kch_100. Similarly BS$St_Kch_101 belongs to records of year 2001 and so on

Student Data for Karachi ***(Cont.)***

- **Step 5b** ***(Cont.)*** **:**
  - For Ms-Excel, it is a convention to use '$' after worksheet name like 'BS$' / 'MS$'
  - If there are any logical divisions of records within an Excel worksheet, it is written after '$' sign like 'BS$St_Kch_94'
  - In the given dataset of Karachi BS Excel worksheet contains data records for eleven years (94–04)and MS Excel worksheet contains data records for 4 years (01-04)
  - Records with in an Excel worksheet are logically divided on annual basis hence we can see options like 'BS$ST_Kch_94'

For MS-Excel, it is a convention to use '$' after worksheet name like 'BS$' / 'MS$'. If there are any logical divisions of records within an Excel worksheet, it is written after '$' sign like 'BS$St_Kch_94'. In the given dataset of Karachi BS Excel worksheet contains data records for eleven years (94–04)and MS Excel worksheet contains data records for 4 years (01-04). Records with in an Excel worksheet are logically divided on annual basis hence we can see options like 'BS$ST_Kch_94'.

Student Data for Karachi ***(Cont.)***

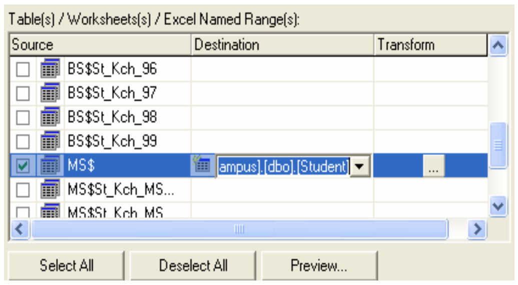
- **Step 5b** ***(Cont.)*** **:**
  - As we want to copy all records from BS worksheet and all records from MS

worksheet as well, therefore, we will choose two options from sources BS$ and MS$

- As Records from both worksheets are required to be copied into a single table Student, therefore, rename destinations against both sources as "Students"

Loading data from MS-Excel workbook provides us facility of loading data in all worksheet through single package. On the same dialog box, scroll down and find option MS$ under the column Source. By this way you can load other worksheet MS in the same table. Locate the row having check box against MS$ and rename the destination as 'Student'. So that all data from both worksheets load in the same table Karachi.

Student Data for Karachi *(Cont.)*



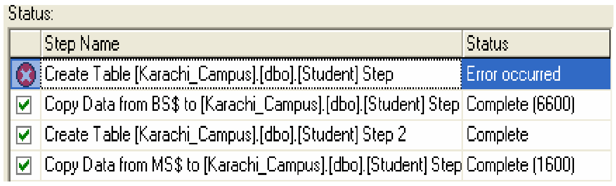
Table(s) / Worksheets(s) / Excel Named Range(s):

	Source	Destination	Transform
☐	BS$St_Kch_96		
☐	BS$St_Kch_97		
☐	BS$St_Kch_98		
☐	BS$St_Kch_99		
✓	MS$	ampus].[dbo].[Student]	...
☐	MS$St_Kch_MS...		
☐	MS$St_Kch_MS		

So, finally two options should be checked under the column Source, one is BS$ and the other one is MS$. Against both options the table name should be the same i.e. Student, because our destination is same. We can use third column transform, by pressing button against the selected row. This option provides us the same mapping view that we have seen earlier, while loading text file.

Student Data for Karachi *(Cont.)*

- **Step 6:**



Status:

	Step Name	Status
✗	Create Table [Karachi_Campus].[dbo].[Student] Step	Error occurred
✓	Copy Data from BS$ to [Karachi_Campus].[dbo].[Student] Step	Complete (6600)
✓	Create Table [Karachi_Campus].[dbo].[Student] Step 2	Complete
✓	Copy Data from MS$ to [Karachi_Campus].[dbo].[Student] Step	Complete (1600)

- Run the package immediately
- Status after package completion is as follows:

After this, run the package. After successful execution we can see the above window. We can see that there were four steps in this case:

1) Create table student
2) Load data of Karachi BS students

3) Create table Student
4) Load data of Karachi MS students

As we checked two options therefore package will try to create the two destination tables. But both tables have same names therefore SQL Server database will not allow to create two tables with the same name 'Student'. Hence out of four, Task three will be terminated with an error '*Student table already exists'*. As no new destination table is created for MS student therefore finally all records of MS worksheet will be loaded in the same table 'Student', due to same destination name. Same information can be seen in above dialog, but these errors are not reported in the order in which they occur. Therefore we can see first task failed where as it is third in reality.

Student Data for Karachi *(Cont.)*

- **Step 6 *(Cont.)*:**
    - It can be seen that a task listed first in previous figure is failed
    - Choosing Source 'BS$' and selecting destination table 'Student' means create table student and copy all records from destination worksheet to student table
    - Similarly Choosing source 'MS$' and selecting destination table 'Student' means exactly the same
    - Hence package tries to create table student twice, once for BS and then for MS

It can be seen that a task listed first in previous run failed. Choosing Source 'BS$' and selecting destination table 'Student' means create table student and copy all records from destination worksheet to student table. Similarly Choosing source 'MS$' and selecting destination table 'Student' means exactly the same. Hence package tries to create table student twice, once for BS and then for MS

Student Data for Karachi *(Cont.)*

- **Step 6 *(Cont.)*:**
    - But second time SQL does not allow to create the table with the same name
    - Therefore a task to create table again fails
    - As there exists only one table student therefore all data for BS and MS is copied to the same table
    - Rest of the three tasks are successful
        - Create table student
        - Copy BS records worksheet
        - Copy MS records worksheet

But second time SQL does not allow to create the table with the same name. Therefore a task to create table again fails. As there exists only one table student therefore all data for BS and MS is copied to the same table. Rest of the three tasks are successful:
1. Create table student,
2. Copy BS records worksheet &
3. Copy MS records worksheet

However, their order of reporting is not the same in which they actually occurred.

Registration Data for Karachi

- Similarly load Registration data for Karachi Campus
- There are two Excel workbooks for Registration data of Karachi
  - Reg_BS_KHR
  - Reg_MS_KHR
- Reg_BS_KHR contains six worksheets each containing registration records of two batches
- Reg_MS_KHR contains only single worksheet contianing all records of 4 batches

Now we are required to load registration detailed records for Karachi campus. There are two MS-Excel workbooks, one for BS & the other one for MS. Workbook for BS contains 6 worksheets, each sheet containing records for 2 batches whereas the worksheet for MS contains all records in one worksheet. We need to load all worksheets complete.

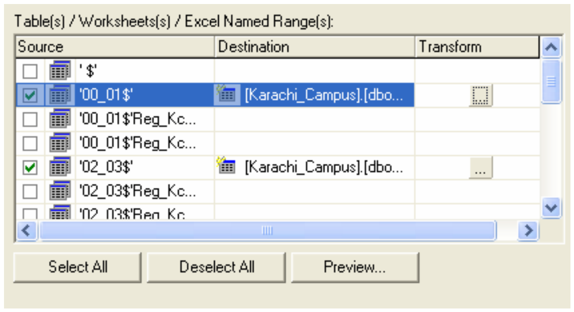
Registration Data for Karachi *(Cont.)*

- First of all load all BS records into a new table 'Registration' in database 'Karachi_Campus'
- Then append all Ms records into the same table 'Registration'
- All these tasks can be performed through wizard in the same way as previous data was loaded

For loading all BS records in a new table 'Registration' we can write only one package. Then we will develop another package that will append all MS records in Registration table. This all can be done in the similar way as we have done before.

To select complete worksheet to be copied we need to check the row indicating the name of worksheet followed by immediate '$' without any logical division. Then set destination table as registration. All registration records either for BS or for MS should be loaded in the same registration table.

Registration Data for Karachi *(Cont.)*


Table(s) / Worksheets(s) / Excel Named Range(s):

Source	Destination	Transform
☐ ' $'		
✓ '00_01$'	[Karachi_Campus].[dbo...	...
☐ '00_01$'Reg_Kc...		
☐ '00_01$'Reg_Kc...		
✓ '02_03$'	[Karachi_Campus].[dbo...	...
☐ '02_03$'Reg_Kc...		
☐ '02_03$'Reg_Kc		

The above slide shows that we are selecting only those options that are showing name of worksheet followed by '$; sign. Against both options we are setting destination as 'Registration'.

Registration Data for Karachi ***(Cont.)***

- As the source is again an Excel book t herefore we have sources with same convention i.e. Worksheet name followed by '$' and logical division of work sheet

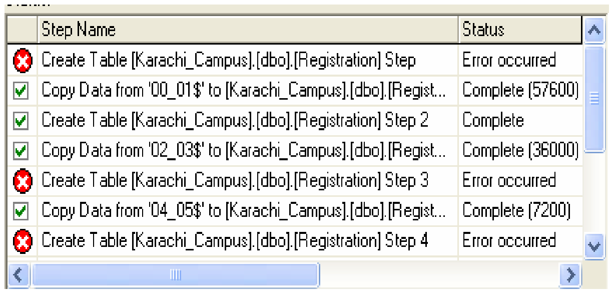
- Check only those sources that corresponds to full worksheet like '94-95$', such sources always miss the name of logical division after '$' sign

As the source is again an Excel book therefore we have sources with same convention i.e. Worksheet name followed by '$' and logical division of work sheet. Check only those sources that correspond to full worksheet like '94-95$', such sources always miss the name of logical division after '$' sign.

Checking each of the check boxes, results in two tasks in the final package. One is to create a new table, always, with the name specified in destination with the same structure as specified by the source table. As we have set same name for destination table that is Registration, therefore only one task of new table creation will succeed rest of all Create Table task will fail and data will be loaded in the same table Registration.

Registration Data for Karachi ***(Cont.)***

- Status of completion is as follows



	Step Name	Status
✗	Create Table [Karachi_Campus].[dbo].[Registration] Step	Error occurred
✓	Copy Data from '00_01$' to [Karachi_Campus].[dbo].[Regist...	Complete (57600)
✓	Create Table [Karachi_Campus].[dbo].[Registration] Step 2	Complete
✓	Copy Data from '02_03$' to [Karachi_Campus].[dbo].[Regist...	Complete (36000)
✗	Create Table [Karachi_Campus].[dbo].[Registration] Step 3	Error occurred
✓	Copy Data from '04_05$' to [Karachi_Campus].[dbo].[Regist...	Complete (7200)
✗	Create Table [Karachi_Campus].[dbo].[Registration] Step 4	Error occurred

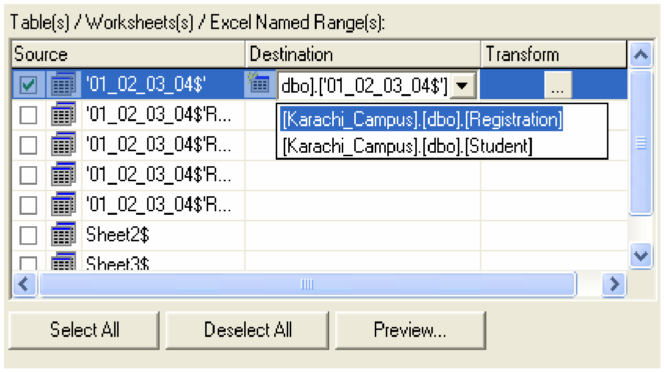
This time it can be easily identified that only those tasks are failed that attempted to create a new table Registration. Rest of all tasks of copy data copied data into same destination table.

Registration Data for Karachi ***(Cont.)***

- All tasks intended to create a new table with the same name 'Registration' are failed
- All the records are appended to already created single table 'Registration'
- Similarly append the Records for MS students in the same Registration table

All tasks intended to create a new table with the same name ‘Registration’ are failed. All the records are appended to the already created single table ‘Registration’. Similarly append the Records for MSstudents in the same Registration table

Registration Data for Karachi *(Cont.)*



Source	Destination	Transform
✓ '01_02_03_04$'	dbo].['01_02_03_04$']	...
'01_02_03_04$'R...	[Karachi_Campus].[dbo].[Registration]	
'01_02_03_04$'R...	[Karachi_Campus].[dbo].[Student]	
'01_02_03_04$'R...		
'01_02_03_04$'R...		
Sheet2$		
Sheet3$		

Similarly load MS data from the workbook **Reg_MS_KHR.** Again set Registration table as destination table and perform remaining steps as it is. As Registration table already exists, again create table task would fail and data will be loaded in the same Registration table, appending new records.

Demographics (1)

- Finally all the data for Karachi Campus has been loaded
- Now we can collect demographics on this data through queries
- First we try the same queries that we used for Lahore Campus

Finally all the data for Karachi campus has been loaded. For collection of demographics, first of all we try the same queries that we used for Lahore campus. As the table names are same i.e. Student and Registration, therefore, those queries should work here also.

Demographics (2)

- **Total Number of Students**

```
Select COUNT(*) AS Expr1
From Student
```

- **8,200**
- **Total BS students**

```
SELECT   COUNT(*) AS Expr1
FROM     Student
WHERE    ([Last Degree] IN ('F.Sc.', 'FSc', 'A level', 'A -Level', 'HSSC'))
```

- Error, [Last Degree] invalid column

Total Number of Students

Select COUNT(*) AS Expr1 From Student;
This query returns 8,200, which is correct. This is exactly the same query that we have already run for Lahore campus. Due to the standardization of table names we get this benefit.

Total BS students

```
  SELECT     COUNT(*) AS Expr1
  FROM        Student
  WHERE     ([Last Degree] IN ('F.Sc.', 'FSc', 'A level', 'A -Level', 'HSSC'))
```

This query returns **Error, [Last Degree] invalid column.** Because in Karachi campus student table, there is no column name as [Last Degree], but same information is stored in qualification column.

Demographics (3)

- At Karachi campus, correct column is 'Qualification' instead of [Last Degree], similarly names of others columns are also different from those of Lahore, therefore, same queries are not applicable here
- **Correct Query to count BS students is**

```
    SELECT     COUNT(*) AS Expr1
    FROM        Student
    WHERE     (Qualification IN ('F.Sc.', 'FSc', 'A level', 'A -Level', 'HSSC'))
```

- **6,600**

This is one of the characteristic of non standardized and heterogeneous data that same information is stored under different column name. Therefore, we need to standardize all column names and data-types so that data from different databases can be put together. Now, to extract same information for Karachi campus we need to correct the above query as

```
SELECT     COUNT(*) AS Expr1
FROM        Student
WHERE    (Qualification IN ('F.Sc.', 'FSc', 'A level', 'A-Level', 'HSSC'))
```

Now there is no problem and query gives correct results (6,600) for Karachi campus. There may be some outliers but that can not be identified at this stage.

Demographics (4)

- To find total number of male students we use same query that we used for Lahore except Column name 'Gender' which is 'M/F' here, so query is

```
SELECT     COUNT(*) AS Expr1
FROM        Student
WHERE     ([M/F] = '0')
```

- **0, Incorrect answer**
- Answer to the query is zero i.e. no male student in Karachi campus which is definitely incorrect

Now we try to find the number of male students at Karachi campus. We submit the same SQL query that we run for Lahore campus. The answer turns out to be zero. There must be some thing incorrect in this query otherwise it can not be zero. To identify the error we look at the data and can easily point out that the error is due to the inconsistent gender storage conventions. At Lahore campus, 1 is a convention to store female and 0 is a convention to store male whereas at Karachi campus M for male and F for female is used. To put all data in single source we need to standardize the data storage conventions.

Demographics (5)

- If we look in the data for Karachi campus we can find out the reason for the error in above query
- For Karachi campus convention to store gender information is M (male) and F (female) instead of 0 (male) and 1 (female)
- **Correct query is**

```
SELECT     COUNT(*) AS Expr1
FROM        Student
WHERE     ([M/F] = 'M')
```

- **5,463**

Following is the correct query that can be used for Karachi campus to count the total number of male students at Karachi campus.

```
SELECT     COUNT(*) AS Expr1
FROM        Student
WHERE     ([M/F] = 'M')
```

The answers to be 5,463.

Demographics (6)

- Total Number of Female students in BS Telecom

- We again need to correct the query which was written for Lahore Campus, and correct query is

```
                    SELECT     COUNT(DISTINCT r.St_ID) AS Expr1
                    FROM     Registration r INNER JOIN
                                 Student s ON r.St_ID = s.St_ID
                                 AND s.Qualification IN ('F.Sc.', 'FSc',
                               ‘HSSC', 'A-Level', 'A level') AND r.Disp = 'TC'
                    AND s.[M/F] = 'F'
```

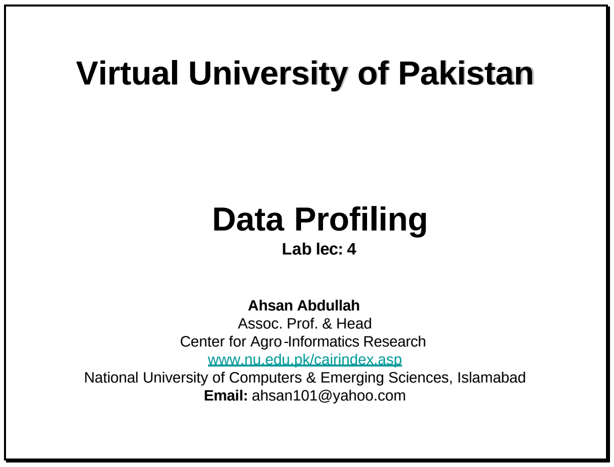
- 551

Similarly to get the count of total number of female students in BS telecom, we need to correct the column names and data storage conventions. The correct query is as follows:

```
SELECT     COUNT(DISTINCT r.St_ID) AS Expr1
FROM     Registration r INNER JOIN
             Student s ON r.St_ID = s.St_ID
             AND s.Qualification IN ('F.Sc.', 'FSc',‘HSSC', 'A-Level', 'A level') AND
r.Disp = 'TC'
            AND s.[M/F] = 'F'
```

Answer is 551

Slide 1

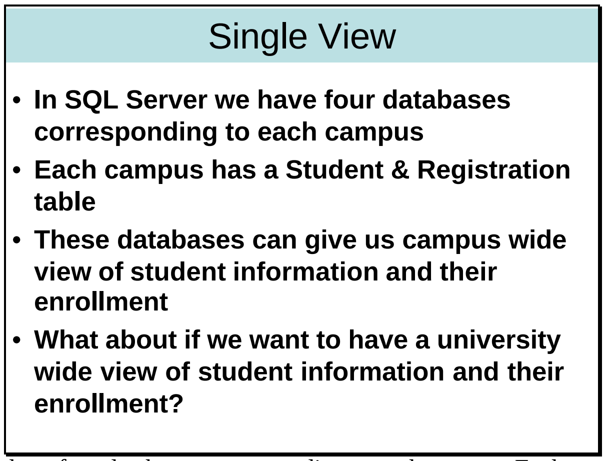

Virtual University of Pakistan

Data Profiling

Lab lec: 4

Ahsan Abdullah
Assoc. Prof. & Head
Center for Agro-Informatics Research
www.nu.edu.pk/cairindex.asp
National University of Computers & Emerging Sciences, Islamabad
Email: ahsan101@yahoo.com

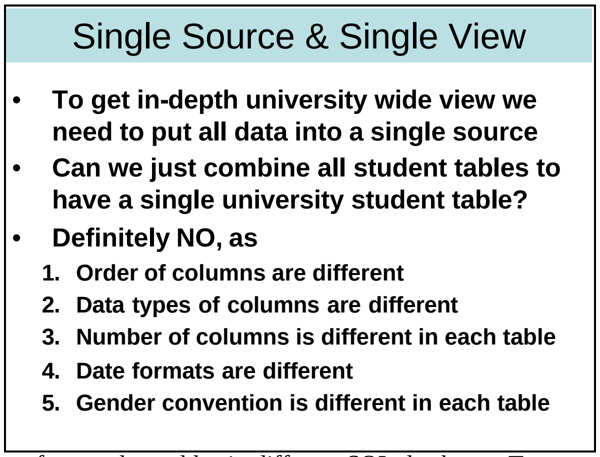
Slide 2


Single View

- **In SQL Server we have four databases corresponding to each campus**
- **Each campus has a Student & Registration table**
- **These databases can give us campus wide view of student information and their enrollment**
- **What about if we want to have a university wide view of student information and their enrollment?**

In SQL Server we have four databases corresponding to each campus. Each campus has a Student & Registration table. Now both tables for each campus have been loaded to MS SQL sever. It means by this stage we have four databases (one for each campus) and corresponding to each database we have two tables.
These databases can give us campus wide view of student information and their enrollment. In order to get university wide view of information we have to have consolidated information of all campuses. To consolidate we need to standardize information across all campuses. In this lecture we will step towards consolidation.

Slide 3

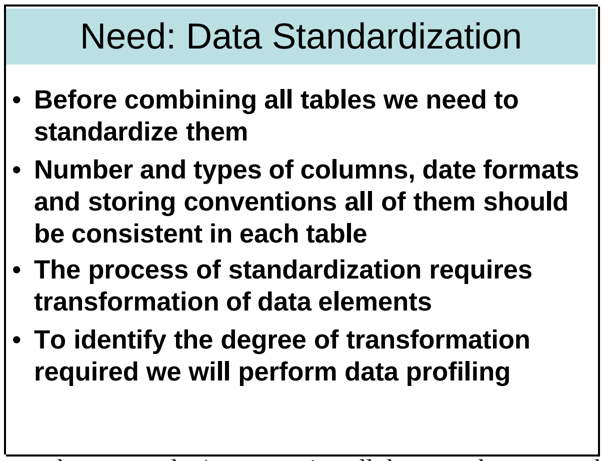


Single Source & Single View

- To get in-depth university wide view we need to put all data into a single source
- Can we just combine all student tables to have a single university student table?
- Definitely NO, as
  1. Order of columns are different
  2. Data types of columns are different
  3. Number of columns is different in each table
  4. Date formats are different
  5. Gender convention is different in each table

At this time we have four student tables in different SQL databases. To consolidate all tables to get single source of truth we can not just glue all tables because of following facts:
Order of columns are different
Data types of columns are different
Number of columns is different in each table
Date formats are different
Gender convention is different in each table

Slide 4

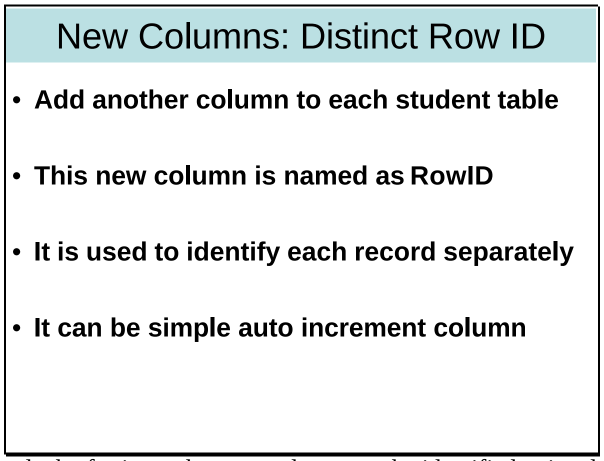


Need: Data Standardization

- Before combining all tables we need to standardize them
- Number and types of columns, date formats and storing conventions all of them should be consistent in each table
- The process of standardization requires transformation of data elements
- To identify the degree of transformation required we will perform data profiling

To remove the factors that are not letting us putting all data together we need to standardize all tables. Standardization process involves the consistency of number and types of columns, date formats, and storing conventions across all campuses and more. To standardize we need to transform data elements. To identify the degree of transformation required we need to perform data profiling.

Slide 5

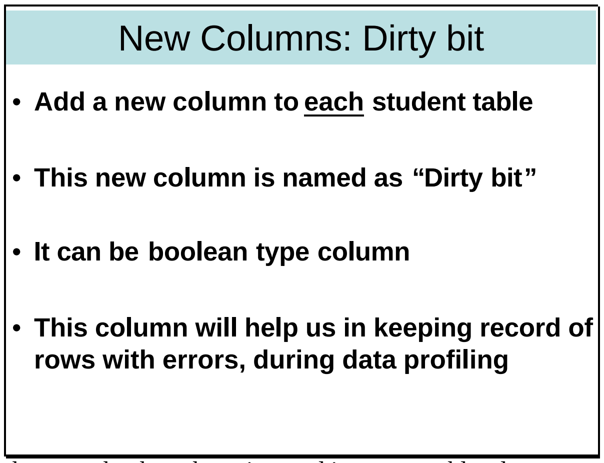


New Columns: Distinct Row ID

- **Add another column to each student table**
- **This new column is named as RowID**
- **It is used to identify each record separately**
- **It can be simple auto increment column**

By this time, due to lack of primary key, records can not be identified uniquely. At this stage we need an attribute that can identify each record uniquely. We need such an identifying attribute at this stage because a lot of records in this stage will be put to error or exception table due to error in any of the columns. In the error table they will be corrected and later on after correction the changes will be updated in original student table. For this update we need a column like row id to join error table with student table.

Slide 6

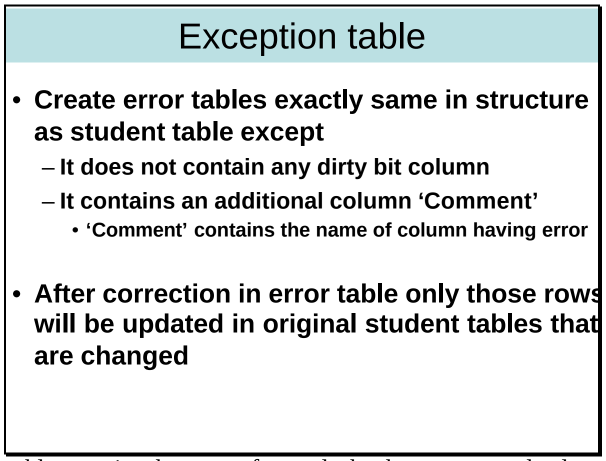


New Columns: Dirty bit

- **Add a new column to each student table**
- **This new column is named as "Dirty bit"**
- **It can be boolean type column**
- **This column will help us in keeping record of rows with errors, during data profiling**

To keep record of the rows that have been inserted into error tables due to certain errors we need an additional column in student table that will serve as dirty bit. Dirty bit of those records is set to true that are inserted in the error table.

Slide 7

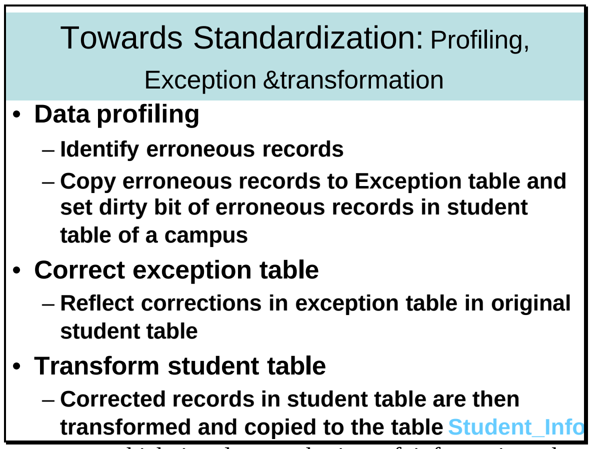


Exception table

- Create error tables exactly same in structure as student table except
  - It does not contain any dirty bit column
  - It contains an additional column 'Comment'
    - 'Comment' contains the name of column having error
- After correction in error table only those rows will be updated in original student tables that are changed

Error or exception table contains the copy of records that have corrupted values for any column in original student table. Error table is the copy of original student table except instead of dirty bit we will have comments column in the error table. In comments column we store the name of columns in which we encountered errors for this particular row. For example consider a record 'R', it has missing gender and incorrect date of birth. For the record R we will have comment **[Gender], [Date of Birth].** These comments will be used while correction of error tables. After correction of error table we will update corresponding rows in the original student table.

Slide 8



Towards Standardization: Profiling, Exception &transformation

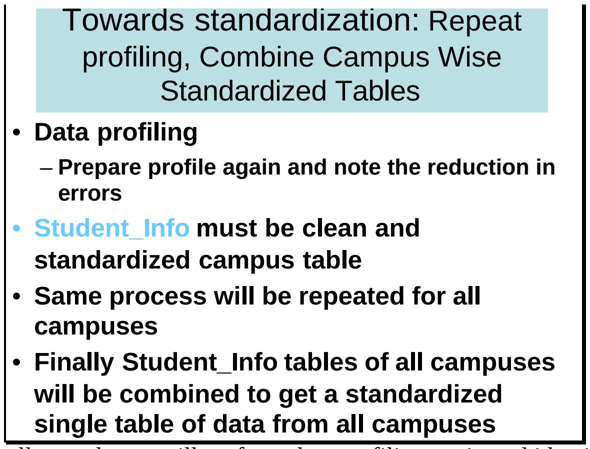
- Data profiling
  - Identify erroneous records
  - Copy erroneous records to Exception table and set dirty bit of erroneous records in student table of a campus
- Correct exception table
  - Reflect corrections in exception table in original student table
- Transform student table
  - Corrected records in student table are then transformed and copied to the table Student_Info

Data profiling is a process which involves gathering of information about column through execution of certain queries with intention to identify erroneous records. In this process we identify the following:

Total number of values in a column
Number of distinct values in a column
Domain of a column
Values out of domain of a column
Validation of business rules

We run different SQL queries to get the answers of above questions. During this process we can identify the erroneous records. Whenever we will come across an erroneous record we will just copy it in error or exception table and set the dirty bit of record in the actual student table.
Then we will correct the exception table. After this profiling process we will transform the records and load them into a new table Student_Info.

Slide 9

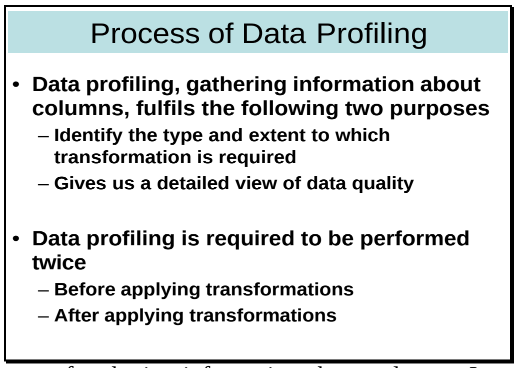


Towards standardization: Repeat profiling, Combine Campus Wise Standardized Tables

- **Data profiling**
  - **Prepare profile again and note the reduction in errors**
- **Student_Info must be clean and standardized campus table**
- **Same process will be repeated for all campuses**
- **Finally Student_Info tables of all campuses will be combined to get a standardized single table of data from all campuses**

After transforming all records, we will perform data profiling again and identify the reduction in the number of errors.
Student_Info must be clean and standardized campus table. After performing same steps for all campuses we will just put Student_Info tables from four campuses together to get consolidated source.

Slide 10



Process of Data Profiling

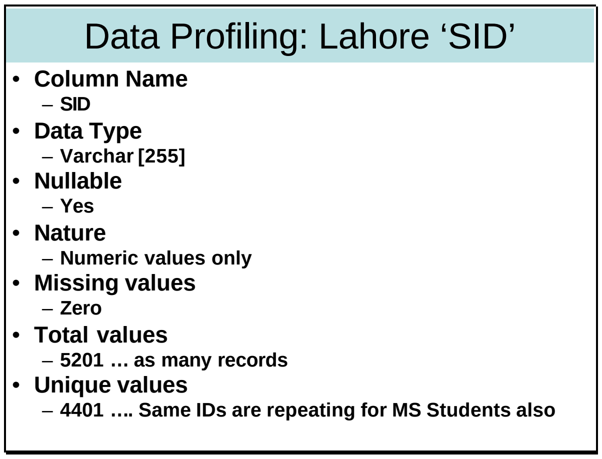
- **Data profiling, gathering information about columns, fulfils the following two purposes**
  - **Identify the type and extent to which transformation is required**
  - **Gives us a detailed view of data quality**
- **Data profiling is required to be performed twice**
  - **Before applying transformations**
  - **After applying transformations**

Data profiling is a process of gathering information about columns, It must fulfil the following purposes

- Identify the type and extent to which the transformation is required
- The number of columns which are required to be transformed and which transformation is required, meaning date format or gender convention.
- It should provide us a detailed view about the quality of data. The number of erroneous values and the number of values out of domain.

To judge effectiveness of transformation we perform data profiling twice. One before transformation and the other after transformation.

Slide 11

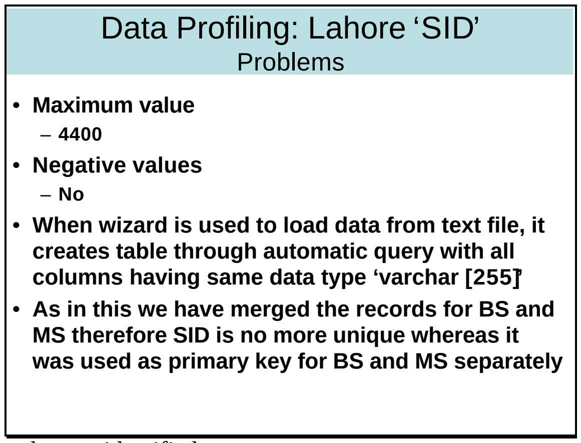


Data Profiling: Lahore 'SID'

- **Column Name**
  - **SID**
- **Data Type**
  - **Varchar [255]**
- **Nullable**
  - **Yes**
- **Nature**
  - **Numeric values only**
- **Missing values**
  - **Zero**
- **Total values**
  - **5201 … as many records**
- **Unique values**
  - **4401 …. Same IDs are repeating for MS Students also**

The slide shows data profiling for Student ID. The results can help us in identifying that what transformations should be applied. Like we can see from here that data type is highly incompatible. To store SID we should not have varchar[255], we can use varchar[10] at maximum.
To know whether column is unique or not, we can compare total values with distinct / unique values. If both values are same then its mean that our column is unique. Similarly we can also identify and count the total number of null values.

Slide 12



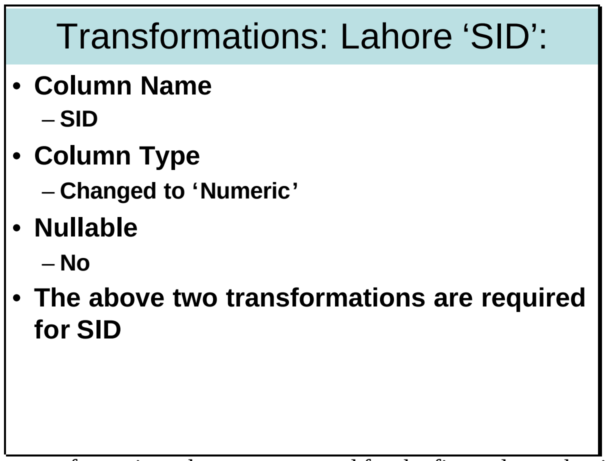
Data Profiling: Lahore 'SID'
Problems

- **Maximum value**
  - **4400**
- **Negative values**
  - **No**
- **When wizard is used to load data from text file, it creates table through automatic query with all columns having same data type 'varchar [255]'**
- **As in this we have merged the records for BS and MS therefore SID is no more unique whereas it was used as primary key for BS and MS separately**

Two major problems that are identified are:

- Varchar[255] is very large to store IDs, names and even addresses, we must need to change it.
- There is no unique column that can be used as primary key

Slide 13

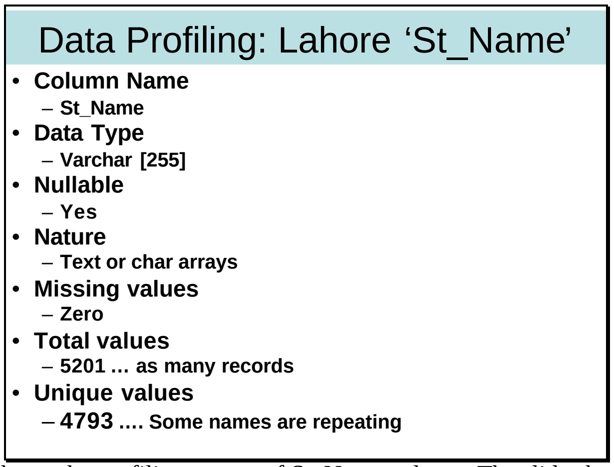


Transformations: Lahore 'SID':

- **Column Name**
  - **SID**
- **Column Type**
  - **Changed to 'Numeric'**
- **Nullable**
  - **No**
- **The above two transformations are required for SID**

The slide shows the transformations that are suggested for the first column that is SID.
First of all its type should be changed to Numeric, as varchar[255] does not make any sense. Secondly we need to change nullability option to no, as ID of each student must be present there. NULLs in ID can cause great trouble while joining tables.

Slide 14

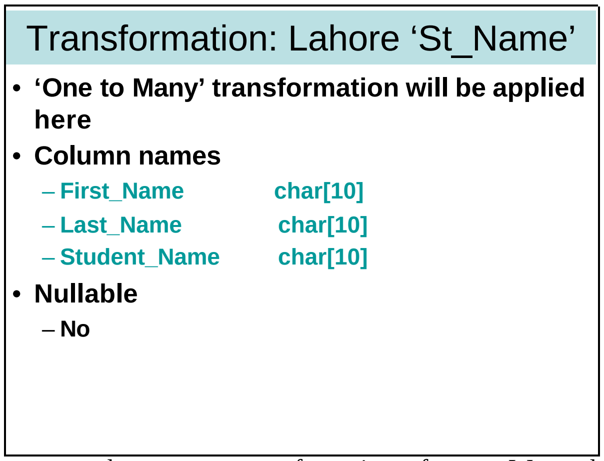


Data Profiling: Lahore 'St_Name'

- **Column Name**
  - **St_Name**
- **Data Type**
  - **Varchar [255]**
- **Nullable**
  - **Yes**
- **Nature**
  - **Text or char arrays**
- **Missing values**
  - **Zero**
- **Total values**
  - **5201 … as many records**
- **Unique values**
  - **4793 …. Some names are repeating**

The above slide shows the profiling output of St_Name column. The slide shows that no names are missing and some of the names are repeating.

Slide 15



Transformation: Lahore 'St_Name'

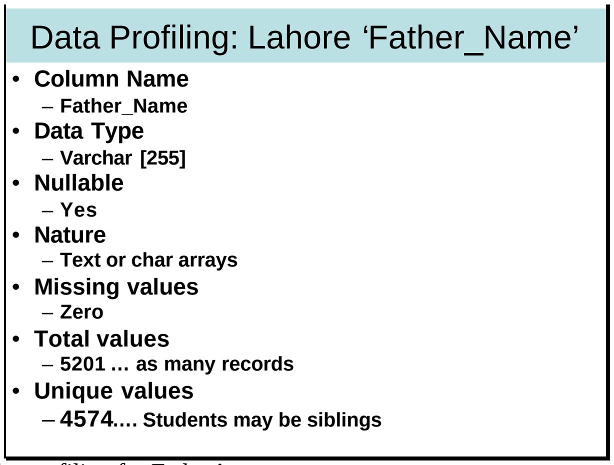
- **'One to Many' transformation will be applied here**
- **Column names**
  - **First_Name char[10]**
  - **Last_Name char[10]**
  - **Student_Name char[10]**
- **Nullable**
  - **No**

To store information we need one to many transformations of names. We need to transform name of each student into 3 columns

- First Name
- Last Name
- Student Name (middle part of name)

This type of transformation requires scripts. We will write VB Scripts for such transformations.

Slide 16

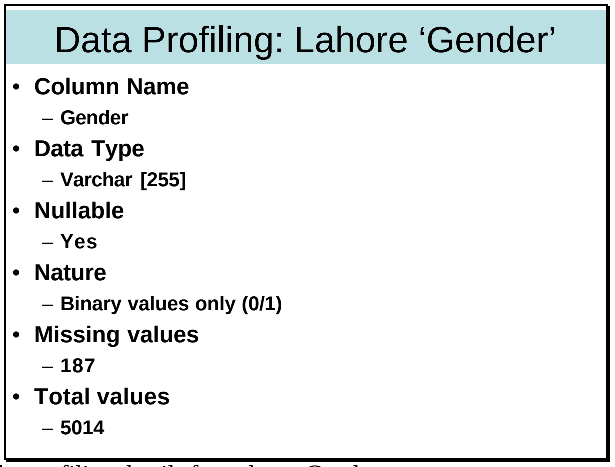


Data Profiling: Lahore 'Father_Name'

- **Column Name**
  - **Father_Name**
- **Data Type**
  - **Varchar [255]**
- **Nullable**
  - **Yes**
- **Nature**
  - **Text or char arrays**
- **Missing values**
  - **Zero**
- **Total values**
  - **5201 … as many records**
- **Unique values**
  - **4574…. Students may be siblings**

The slide shows the profiling for Father's name.

Slide 17

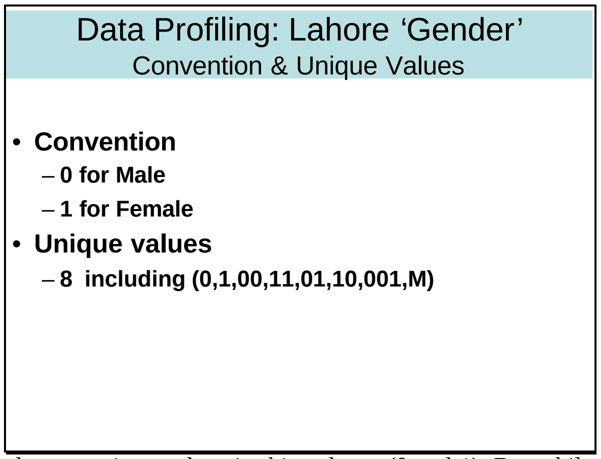


Data Profiling: Lahore 'Gender'

- Column Name
  - Gender
- Data Type
  - Varchar [255]
- Nullable
  - Yes
- Nature
  - Binary values only (0/1)
- Missing values
  - 187
- Total values
  - 5014

The slide shows the profiling details for column Gender.

Slide 18

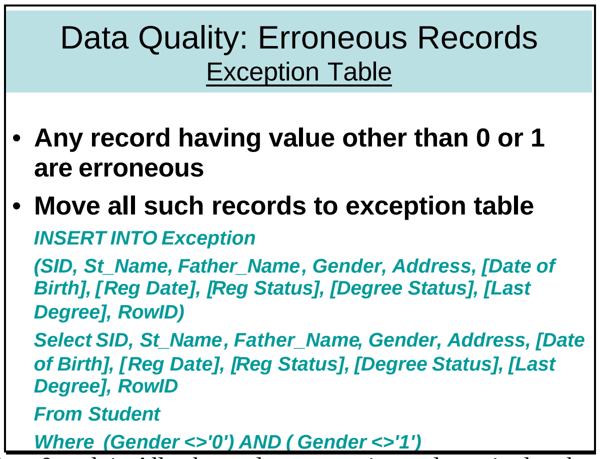


Data Profiling: Lahore 'Gender'
Convention & Unique Values

- Convention
  - 0 for Male
  - 1 for Female
- Unique values
  - 8 including (0,1,00,11,01,10,001,M)

There should be only two unique values in this column (0 and 1). But while profiling we have found that there are 8 unique values. Its mean there are a lot of rows corrupted in original data. Some may be typos or missing values etc. We need to clean data to make it usable for analysis purposes.

Slide 19



Data Quality: Erroneous Records
Exception Table

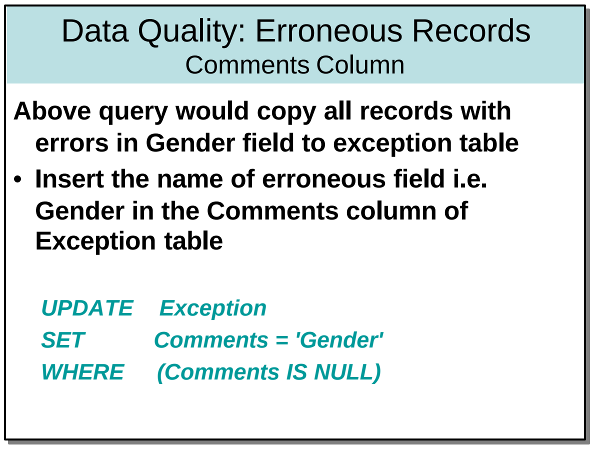
- **Any record having value other than 0 or 1 are erroneous**
- **Move all such records to exception table**

```
INSERT INTO Exception
(SID, St_Name, Father_Name, Gender, Address, [Date of Birth], [Reg Date], [Reg Status], [Degree Status], [Last Degree], RowID)
Select SID, St_Name, Father_Name, Gender, Address, [Date of Birth], [Reg Date], [Reg Status], [Degree Status], [Last Degree], RowID
From Student
Where  (Gender <>'0') AND ( Gender <>'1')
```

Legal values are just 0 and 1. All other values are noise and required to be cleaned. To clean values put all corrupted rows to exception table. Following query can be used to move all corrupted rows to exception table.

INSERT INTO Exception
(SID, St_Name, Father_Name, Gender, Address, [Date of Birth], [Reg Date], [Reg Status], [Degree Status], [Last Degree], RowID)
Select SID, St_Name, Father_Name, Gender, Address, [Date of Birth], [Reg Date], [Reg Status], [Degree Status], [Last Degree], RowID
From Student
Where (Gender <>'0') AND (Gender <>'1')

Slide 20



Data Quality: Erroneous Records
Comments Column

Above query would copy all records with errors in Gender field to exception table

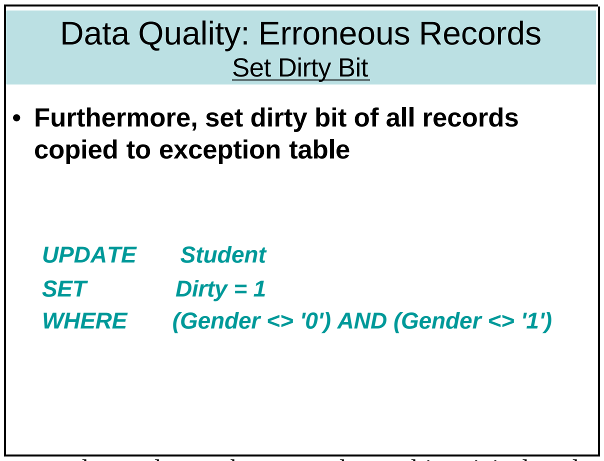
- **Insert the name of erroneous field i.e. Gender in the Comments column of Exception table**

```
UPDATE    Exception
SET       Comments = 'Gender'
WHERE     (Comments IS NULL)
```

As the Comment column was added just to store the name of column having error, therefore, above query is required to be executed to update Comment column to enter the name of corrupted column against each row.

This query is required to be run right after each Insert query to exception table, otherwise we will lose the information 'What was wrong in a particular row?' in exception table.

Slide 21



Data Quality: Erroneous Records

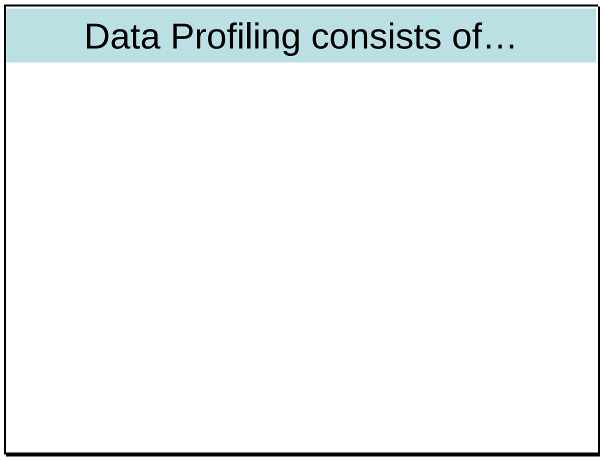
Set Dirty Bit

- **Furthermore, set dirty bit of all records copied to exception table**

```
UPDATE      Student
SET         Dirty = 1
WHERE       (Gender <> '0') AND (Gender <> '1')
```

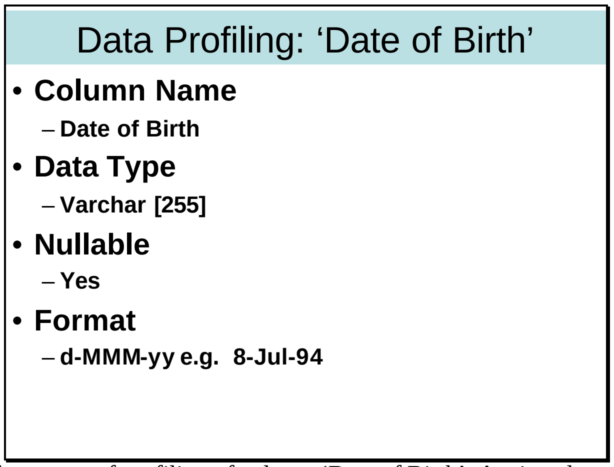
At the same time we need to update each corrupted record in original student table for Lahore campus by setting its dirty bit. So that we can maintain the record which rows are copied in error table. Dirty bit says that this record has at least one field corrupted. It does not show how many fields are corrupted.

Slide 22



Data Profiling consists of…

Slide 23

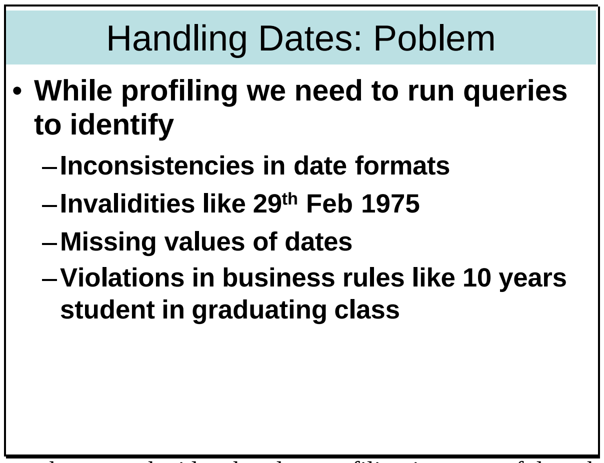


Data Profiling: 'Date of Birth'

- **Column Name**
  - **Date of Birth**
- **Data Type**
  - **Varchar [255]**
- **Nullable**
  - **Yes**
- **Format**
  - **d-MMM-yy e.g. 8-Jul-94**

The slide shows the output of profiling of column 'Date of Birth'. Again column type is incorrect, column can contain null values, and format of date is
d-MMM-yy.
Profiling Date needs to change our flow of work. For columns before this we just did profiling and no transformation has been done yet. We can not profile date without transforming it. So we need to transform date first and then profile. The problem is, when we loaded all records to SQL server from text files they are loaded as strings (character arrays). While profiling we may want to get the range of dates (minimum and maximum dates). We can not identify date ranges until or unless we transform it as date data type.

Slide 24



Handling Dates: Poblem

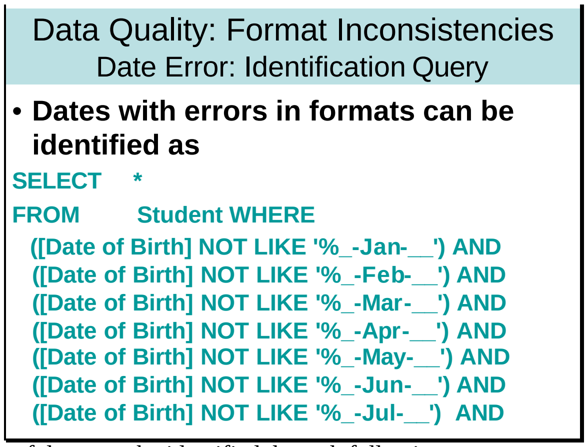
- **While profiling we need to run queries to identify**
  - **Inconsistencies in date formats**
  - **Invalidities like 29th Feb 1975**
  - **Missing values of dates**
  - **Violations in business rules like 10 years student in graduating class**

By this time you must have got the idea that data profiling is a powerful method to have an idea about the quality of data. While profiling data we need to run queries to identify:

- Inconsistencies in date formats
- Invalidities like 29th Feb 1975
- Missing values of dates
- Violations in business rules like 10 years student in graduating class

Not only the values out of domain of columns hit the quality of data but certain values in the domain of column may cause the quality to suffer. Like the values cause the violation of business rules. While profiling data we must have a set of rules regarding to our business like we can not allow any student less than 16 years to be admitted for BS. While profiling we need to identify the records violating such rules means if we find a 10 years old student in graduated students then its the violation of business rule and there must be some error either in date of birth of student or date of graduation. Similarly any student has date of graduation less than date of birth then again it is considered as noise while both dates are in valid domain. Such dates are required to be identified at the time of profiling and needs to be dealt separately in error table.

Slide 25



Data Quality: Format Inconsistencies

Date Error: Identification Query

- **Dates with errors in formats can be identified as**

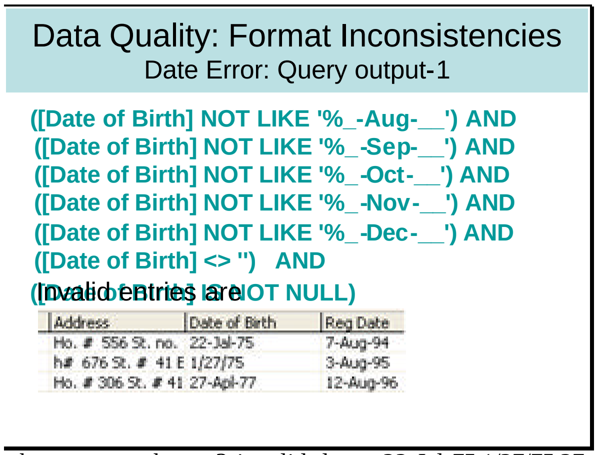
```
SELECT    *
FROM      Student WHERE
  ([Date of Birth] NOT LIKE '%_-Jan-__') AND
  ([Date of Birth] NOT LIKE '%_-Feb-__') AND
  ([Date of Birth] NOT LIKE '%_-Mar-__') AND
  ([Date of Birth] NOT LIKE '%_-Apr-__') AND
  ([Date of Birth] NOT LIKE '%_-May-__') AND
  ([Date of Birth] NOT LIKE '%_-Jun-__') AND
  ([Date of Birth] NOT LIKE '%_-Jul-__')  AND
```

Errors in the format of dates can be identified through following query.

```
SELECT   *
FROM     Student
WHERE    ([Date of Birth] NOT LIKE '%_-Jan-__')
AND      ([Date of Birth] NOT LIKE '%_-Feb-__')
AND      ([Date of Birth] NOT LIKE '%_-Mar-__')
AND      ([Date of Birth] NOT LIKE '%_-Apr-__')
AND      ([Date of Birth] NOT LIKE '%_-May-__')
AND      ([Date of Birth] NOT LIKE '%_-Jun-__')
AND      ([Date of Birth] NOT LIKE '%_-Jul-__')
AND      ([Date of Birth] NOT LIKE '%_-Aug-__')
AND      ([Date of Birth] NOT LIKE '%_-Sep-__')
AND      ([Date of Birth] NOT LIKE '%_-Oct-__')
AND      ([Date of Birth] NOT LIKE '%_-Nov-__')
AND      ([Date of Birth] NOT LIKE '%_-Dec-__')
AND      ([Date of Birth] <> '')   AND      ([Date of Birth] IS NOT NULL)
```

The above query can not identify all invalid dates like 30-Feb-2005. But any date with invalid format, or having illegal month can be identified through the above query.

Slide 26



Data Quality: Format Inconsistencies
Date Error: Query output-1

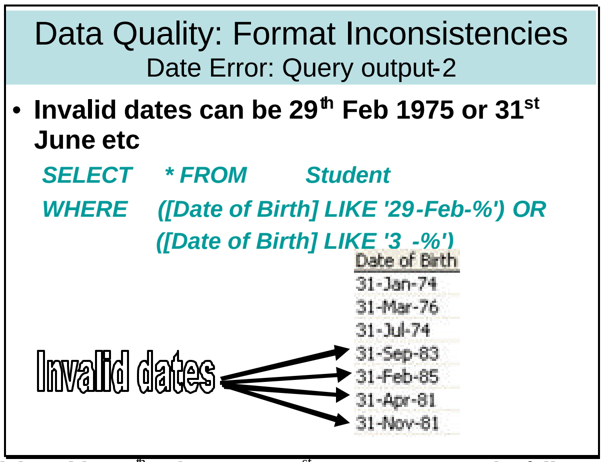
([Date of Birth] NOT LIKE '%_-Aug-__') AND
([Date of Birth] NOT LIKE '%_-Sep-__') AND
([Date of Birth] NOT LIKE '%_-Oct-__') AND
([Date of Birth] NOT LIKE '%_-Nov-__') AND
([Date of Birth] NOT LIKE '%_-Dec-__') AND
([Date of Birth] <> '') AND
([Date of Birth] IS NOT NULL)

Invalid entries are

Address	Date of Birth	Reg Date
Ho. # 556 St. no.	22-Jal-75	7-Aug-94
h# 676 St. # 41 E	1/27/75	3-Aug-95
Ho. # 306 St. # 41	27-Apl-77	12-Aug-96

The output of the above query shows 3 invalid dates. 22-Jal-75,1/27/75,27-Apl-77. These are invalid dates as their format are not correct.
While transforming dates from string to date format, all dates must be legal. If any illegal date comes the package will be terminated and all transactions will be rolled back. So through sequence of different query we try to identify all invalid dates so that our package can complete its execution successfully.

Slide 27



Data Quality: Format Inconsistencies
Date Error: Query output-2

- Invalid dates can be 29th Feb 1975 or 31st June etc

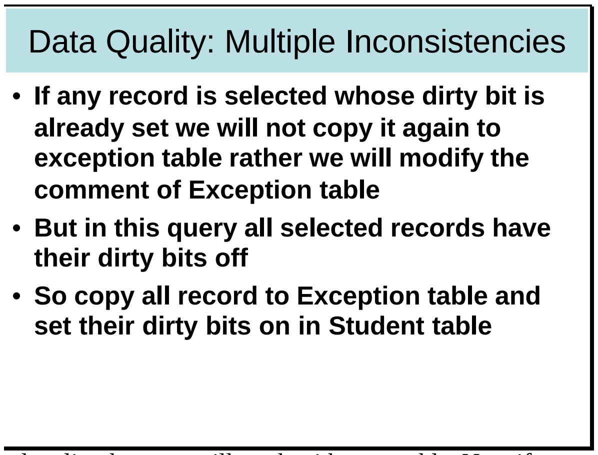
SELECT * FROM Student
WHERE ([Date of Birth] LIKE '29-Feb-%') OR
([Date of Birth] LIKE '3_-%')

Date of Birth
31-Jan-74
31-Mar-76
31-Jul-74
31-Sep-83
31-Feb-85
31-Apr-81
31-Nov-81

Invalid dates

To identify invalid dates like 29th Feb 1975 or 31st June, we can run the following query
SELECT * FROM Student
WHERE ([Date of Birth] LIKE '29-Feb-%') OR
([Date of Birth] LIKE '3_-%')
There is no finalized methodology to identify errors in dates. It is up to the designer what sequence of queries he/she generates to identify the errors in dates because the purpose is to identify all erroneous dates before running the package. Otherwise, package's execution will be terminated and all transactions caused due to that package will be rolled back.

Slide 28

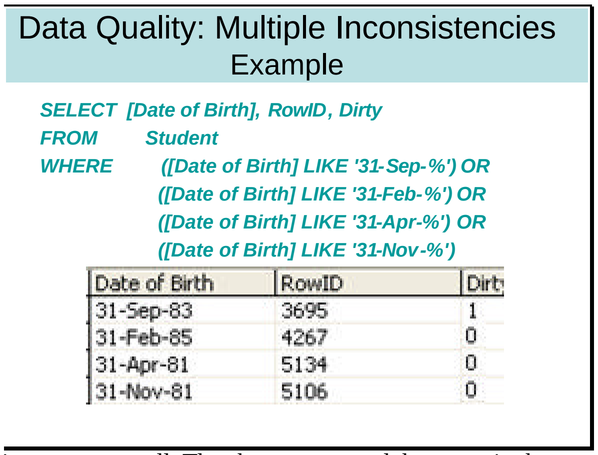


Data Quality: Multiple Inconsistencies

- If any record is selected whose dirty bit is already set we will not copy it again to exception table rather we will modify the comment of Exception table
- But in this query all selected records have their dirty bits off
- So copy all record to Exception table and set their dirty bits on in Student table

It has been discussed earlier, how we will work with error table. Now if we meet a record whose dirty bit is already set then we will not copy the record again but we will modify the comment in exception table. We will append the name of second erroneous column with the name of column already present there.

Slide 29



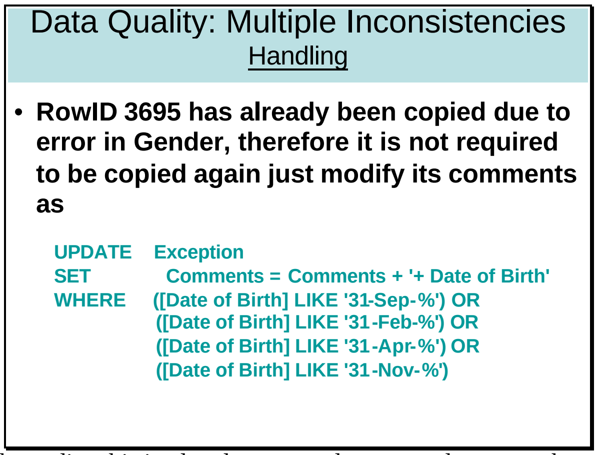
Data Quality: Multiple Inconsistencies Example

```
SELECT  [Date of Birth], RowID, Dirty
FROM        Student
WHERE        ([Date of Birth] LIKE '31-Sep-%') OR
             ([Date of Birth] LIKE '31-Feb-%') OR
             ([Date of Birth] LIKE '31-Apr-%') OR
             ([Date of Birth] LIKE '31-Nov-%')
```

Date of Birth	RowID	Dirt
31-Sep-83	3695	1
31-Feb-85	4267	0
31-Apr-81	5134	0
31-Nov-81	5106	0

We can use following query as well. The above query and the query in the previous slide both are aimed at identify the invalidities in dates. There may be some other set of queries that can be used with the same intention to identify invalid dates. Good queries are those that identify and output only erroneous records and all erroneous records.

Slide 30



Data Quality: Multiple Inconsistencies
Handling

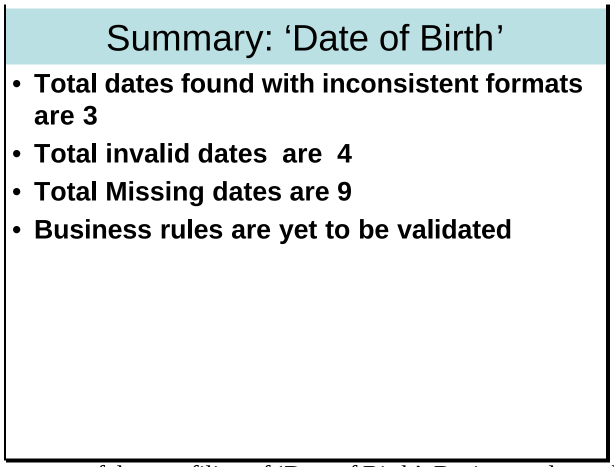
- **RowID 3695 has already been copied due to error in Gender, therefore it is not required to be copied again just modify its comments as**

```
UPDATE    Exception
SET         Comments = Comments + '+ Date of Birth'
WHERE     ([Date of Birth] LIKE '31-Sep-%') OR
           ([Date of Birth] LIKE '31-Feb-%') OR
           ([Date of Birth] LIKE '31-Apr-%') OR
           ([Date of Birth] LIKE '31-Nov-%')
```

For the records whose dirty bit is already on, we do not need to enter the record again but to modify the comment only through following query

```
UPDATE    Exception
SET        Comments = Comments + '+ Date of Birth'
WHERE       ([Date of Birth] LIKE '31-Sep-%') OR
           ([Date of Birth] LIKE '31-Feb-%') OR
           ([Date of Birth] LIKE '31-Apr-%') OR
           ([Date of Birth] LIKE '31-Nov-%')
```

Slide 31

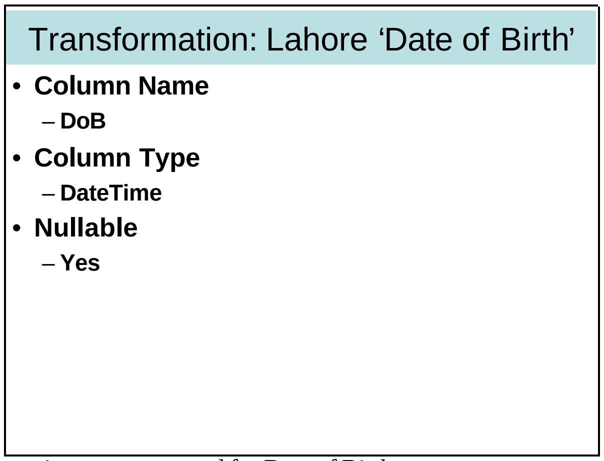


Summary: 'Date of Birth'

- **Total dates found with inconsistent formats are 3**
- **Total invalid dates are 4**
- **Total Missing dates are 9**
- **Business rules are yet to be validated**

Slide shows the summary of data profiling of 'Date of Birth'. Business rule can be as follows:
At the time of joining for BS, student must be at least 16 years old
At the time of joining for MS, student must be at least 20 years old
Business rules are yet to be validated because to validate business rule we have to have date of registration as well.

Slide 32



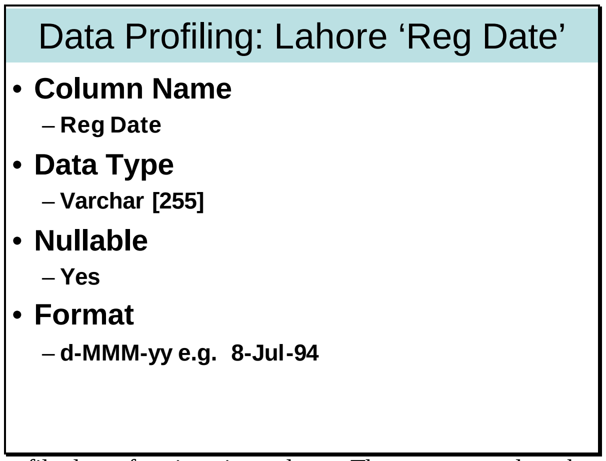
Transformation: Lahore 'Date of Birth'

- **Column Name**
  - **DoB**
- **Column Type**
  - **DateTime**
- **Nullable**
  - **Yes**

Following transformations are suggested for Date of Birth:

- Change the name of column as DoB
- Change column type as DateTime format
- Allow null values as there are some nulls and empty strings in the original data

Slide 33

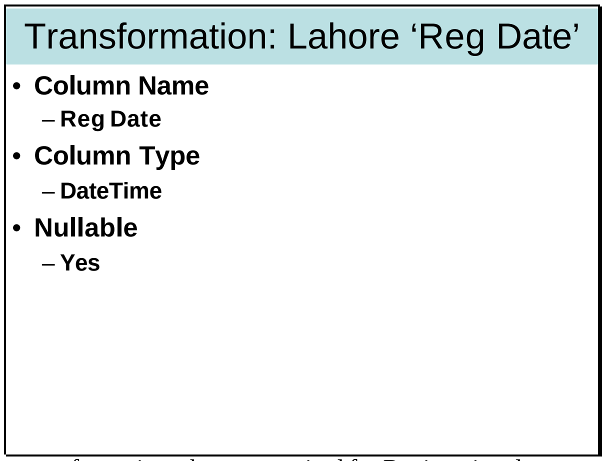


Data Profiling: Lahore 'Reg Date'

- **Column Name**
  - **Reg Date**
- **Data Type**
  - **Varchar [255]**
- **Nullable**
  - **Yes**
- **Format**
  - **d-MMM-yy e.g. 8-Jul-94**

Now we need to profile date of registration column. The same procedure that we have done for Date of Birth is required to be repeated.

Slide 34

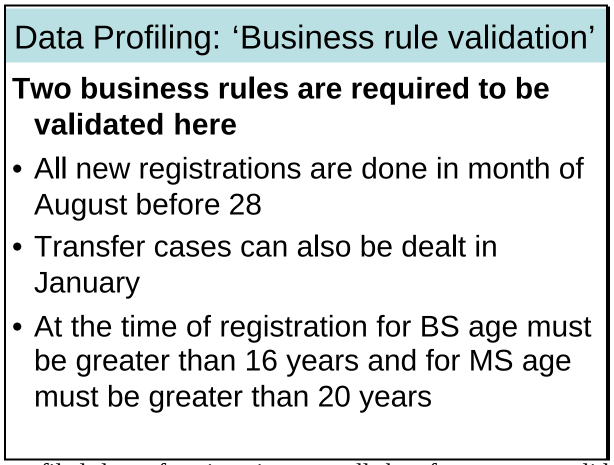


Transformation: Lahore 'Reg Date'

- **Column Name**
  - **Reg Date**
- **Column Type**
  - **DateTime**
- **Nullable**
  - **Yes**

The slide shows the transformations that are required for Registration date.

Slide 35



Data Profiling: 'Business rule validation'

Two business rules are required to be validated here

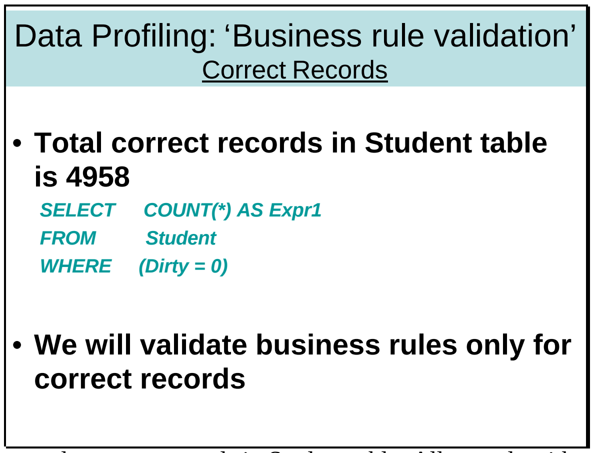
- All new registrations are done in month of August before 28
- Transfer cases can also be dealt in January
- At the time of registration for BS age must be greater than 16 years and for MS age must be greater than 20 years

Now as we have profiled date of registration as well therefore we can validate the following business rules

- All new registrations are done in month of August before 28th
- Transfer cases can also be dealt in January
- At the time of registration for BS age must be greater than 16 years and for MS age must be greater than 20 years

There can be a lot of business rules that are supposed to be validated at this stage. It totally depends upon the business.
Other than business rules we also validate some logical rules at this stage, like date of registration can not be less than date of birth. No one could register for a degree before birth. Similarly date of birth can not be 100 years back and so on. Such rules can be devised keeping in mind the nature of business.

Slide 36



Data Profiling: 'Business rule validation'
Correct Records

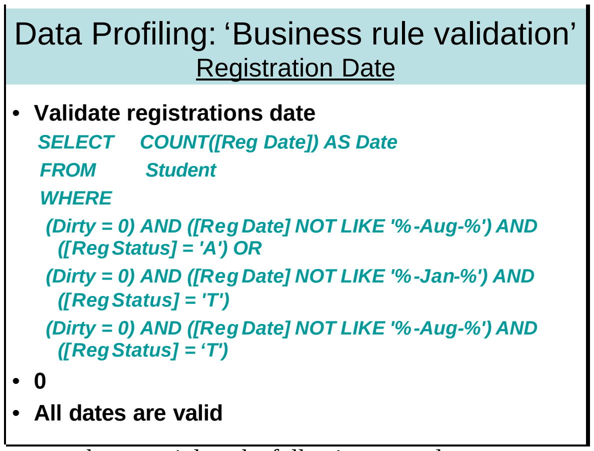
- **Total correct records in Student table is 4958**

```
SELECT    COUNT(*) AS Expr1
FROM      Student
WHERE     (Dirty = 0)
```

- **We will validate business rules only for correct records**

At this stage we have only correct records in Student table. All records with errors in dates have been moved to exception table. Now we will validate the correct records for given set of business rules and general logical rules. Records violating any of the rules are required to copy in exception table.

Slide 37



Data Profiling: 'Business rule validation'
Registration Date

- **Validate registrations date**

```
SELECT    COUNT([Reg Date]) AS Date
FROM      Student
WHERE
 (Dirty = 0) AND ([Reg Date] NOT LIKE '%-Aug-%') AND
   ([Reg Status] = 'A') OR
 (Dirty = 0) AND ([Reg Date] NOT LIKE '%-Jan-%') AND
   ([Reg Status] = 'T')
 (Dirty = 0) AND ([Reg Date] NOT LIKE '%-Aug-%') AND
   ([Reg Status] = 'T')
```

- **0**
- **All dates are valid**

The slide shows the query that can violate the following two rules:

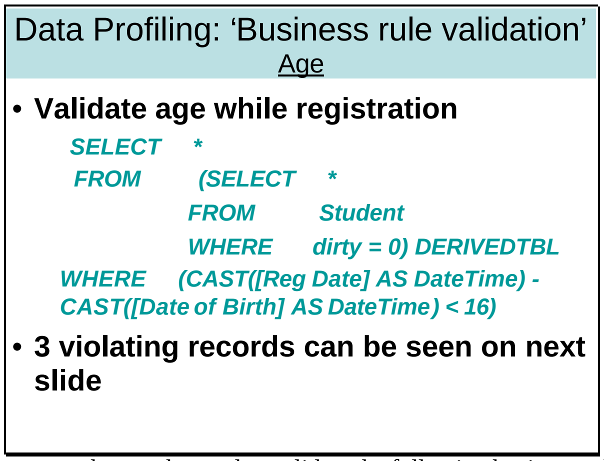
- **All new registrations are done in month of August before 28th**
- **Transfer cases can also be dealt in January**

We can see that the first check ,'dirty bit = 0', is giving us the records having no errors. The following query can identify all records that have registration date in any month other than August and January. Those records will be erroneous records. Records having registration date in January and Registration status is transfer case is legal. Similarly in August both Transfer cases and new admissions are legal.

SELECT   COUNT([Reg Date]) AS Date
FROM   Student
WHERE
(Dirty = 0) AND ([Reg Date] NOT LIKE '%-Aug-%') AND ([Reg Status] = 'A') OR

(Dirty = 0) AND ([Reg Date] NOT LIKE '%-Jan-%') AND ([Reg Status] = 'T')
(Dirty = 0) AND ([Reg Date] NOT LIKE '%-Aug-%') AND ([Reg Status] = 'T')

Slide38



Data Profiling: 'Business rule validation'
Age

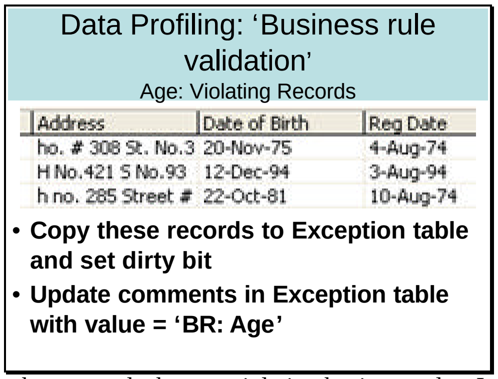
- **Validate age while registration**

```
SELECT   *
FROM     (SELECT   *
          FROM     Student
          WHERE    dirty = 0) DERIVEDTBL
WHERE    (CAST([Reg Date] AS DateTime) -
CAST([Date of Birth] AS DateTime) < 16)
```

- **3 violating records can be seen on next slide**

The slide shows the query that can be used to validate the following business rule:
At the time of registration for BS age must be greater than 16 years and for MS age must be greater than 20 years
The query again first identifies the correct records through dirty bit. Then checks which of the students are less than 16 years at the time of registration. To apply '+' or '-' operator on dates we need to cast date strings into date data type by using CAST function. If there comes any invalid date the query terminates with error.

Slide 39



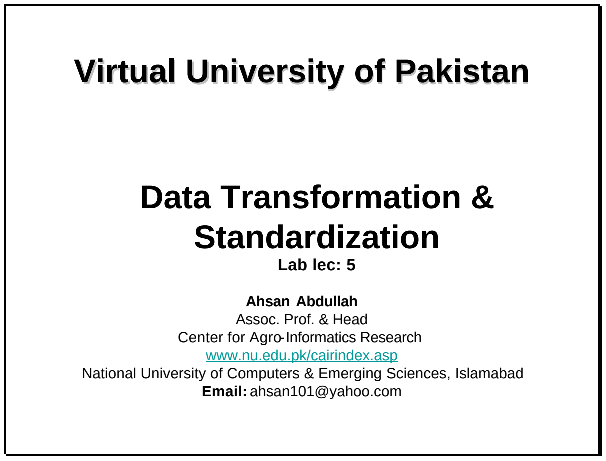
Data Profiling: 'Business rule validation'
Age: Violating Records

Address	Date of Birth	Reg Date
ho. # 308 St. No.3	20-Nov-75	4-Aug-74
H No.421 S No.93	12-Dec-94	3-Aug-94
h no. 285 Street #	22-Oct-81	10-Aug-74

- **Copy these records to Exception table and set dirty bit**
- **Update comments in Exception table with value = 'BR: Age'**

On the slide we can see three records that are violating business rules. In the first record date of birth is 20-Nov-75 and date of Registration is 4-Aug-74, which is obviously impossible. Here, one thing is obvious that date of registration is incorrect as we have data starting from 1994 only. Either date of birth is correct or not, nothing can be said without consulting golden copy (Original copy).
We will copy these records to exception table and set the dirty bit for these records. For such cases we will update comment by appending 'BR:Age' (Business rule: Age).

Slide 1



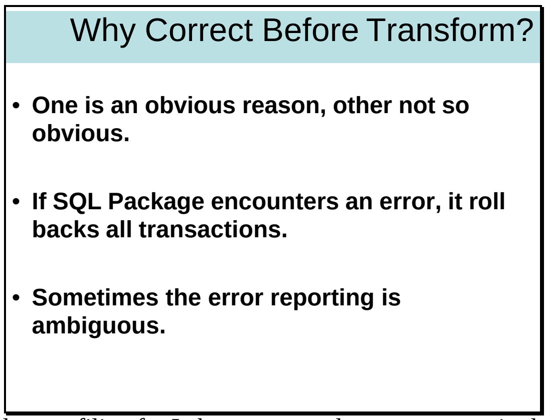
Virtual University of Pakistan

Data Transformation & Standardization

Lab lec: 5

Ahsan Abdullah
Assoc. Prof. & Head
Center for Agro-Informatics Research
www.nu.edu.pk/cairindex.asp
National University of Computers & Emerging Sciences, Islamabad
Email: ahsan101@yahoo.com

Slide 2



Why Correct Before Transform?

- **One is an obvious reason, other not so obvious.**
- **If SQL Package encounters an error, it roll backs all transactions.**
- **Sometimes the error reporting is ambiguous.**

After completion of data profiling for Lahore campus data, we are required to correct the records in the error table. After corrections the corresponding records in the actual table are required to be updated. The methodology to correct the exception table is dependent on the following factors:

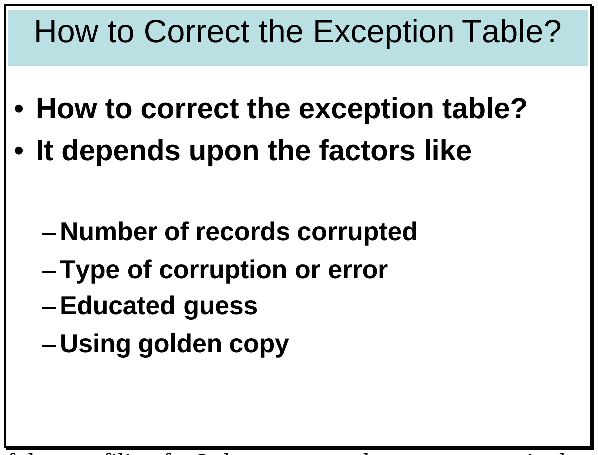
Number of records corrupted

If the number of corrupted records are very large in number then we need to update it through SQL Queries or some other automated way, otherwise, if they are less in number then they can be updated through inspection or manual checking.

Type of corruption or error

If the dates are missing we must need to consult golden copy. If gender is missing we are not required to consult golden copy. In many cases name can help us in identifying the gender of the person.

Slide 3


How to Correct the Exception Table?

- **How to correct the exception table?**
- **It depends upon the factors like**
  - **Number of records corrupted**
  - **Type of corruption or error**
  - **Educated guess**
  - **Using golden copy**

After completion of data profiling for Lahore campus data, we are required to correct the records in the error table. After corrections the corresponding records in the actual table are required to be updated. The methodology to correct the exception table is dependent on the following factors:

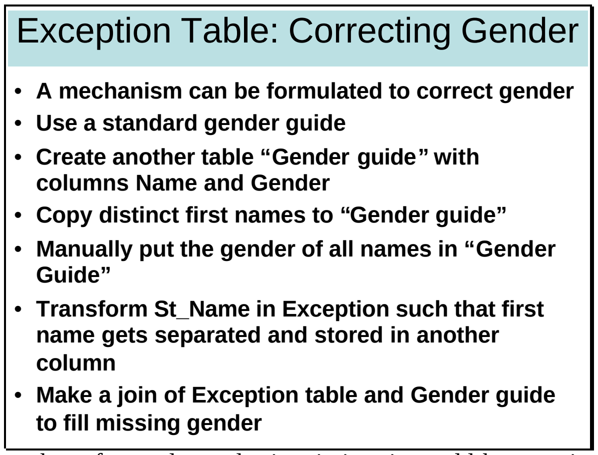
- **Number of records corrupted**

If the number of corrupted records are very large in number then we need to update it through SQL Queries or some other automated way, otherwise, if they are less in number then they can be updated through inspection or manual checking.

- **Type of corruption or error**

If the dates are missing we must need to consult golden copy. If gender is missing we are not required to consult golden copy. Name can help us in identifying the gender of the person.

Slide 4

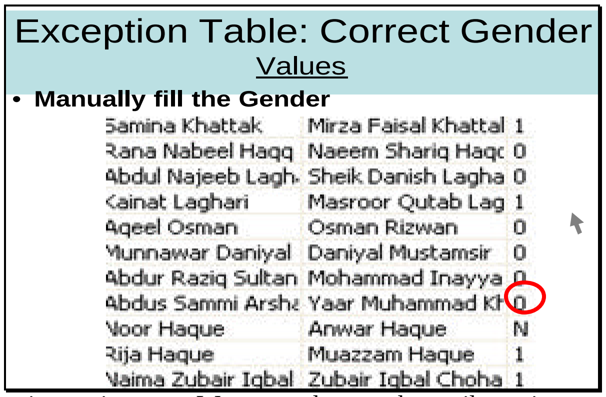

Exception Table: Correcting Gender

- **A mechanism can be formulated to correct gender**
- **Use a standard gender guide**
- **Create another table "Gender guide" with columns Name and Gender**
- **Copy distinct first names to "Gender guide"**
- **Manually put the gender of all names in "Gender Guide"**
- **Transform St_Name in Exception such that first name gets separated and stored in another column**
- **Make a join of Exception table and Gender guide to fill missing gender**

If for very large number of records gender is missing, it would become impossible for us to manually check each and every individual's name and identify the gender. In such cases we can formulate a mechanism to correct gender. We can either use a standard gender guide or create a new table Gender_guide. Gender_guide contains only two columns name and gender. Populate

Gender_guide table by a query for selecting all distinct first names from student table. Then manually placing their gender.
This table can serve us as guide by telling what can be the gender against this particular name. For example if we have hundred students in our database with first name equal to 'Muhammed'. Then in our Gender_guide table we will have just one entry 'Muhammed' and we will manually set the gender as 'Male' against 'Muhammed'.
Now to fill missing genders in exception table we will just do a inner join on Error table and Gender_guide table. We will get the gender against matching names.

Slide 5



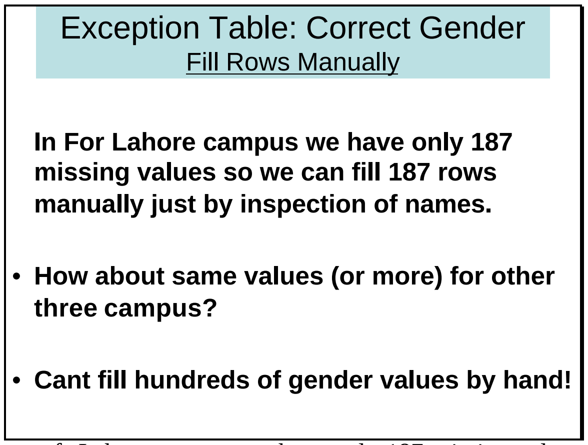
Exception Table: Correct Gender
Values

- **Manually fill the Gender**

Samina Khattak	Mirza Faisal Khattal	1
Rana Nabeel Haqq	Naeem Shariq Haqc	0
Abdul Najeeb Lagh	Sheik Danish Lagha	0
Kainat Laghari	Masroor Qutab Lag	1
Aqeel Osman	Osman Rizwan	0
Munnawar Daniyal	Daniyal Mustamsir	0
Abdur Raziq Sultan	Mohammad Inayya	0
Abdus Sammi Arsha	Yaar Muhammad Kh	0
Noor Haque	Anwar Haque	N
Rija Haque	Muazzam Haque	1
Naima Zubair Iqbal	Zubair Iqbal Choha	1

The slide shows an interesting case. We can update gender easily against names like Sara that clearly identifies that the student is female. But there may be certain names that are common for males and females like Shamim, Khursheed, etc. In the slide we can see a name Noor Haque, doesn't conveying the gender. It may be male and female as well. So, for such case at this stage we can use 'N', but these cases can only be resolved through consulting golden copy. This conversion i.e. N needs to be reflected in meta data.

Slide 6



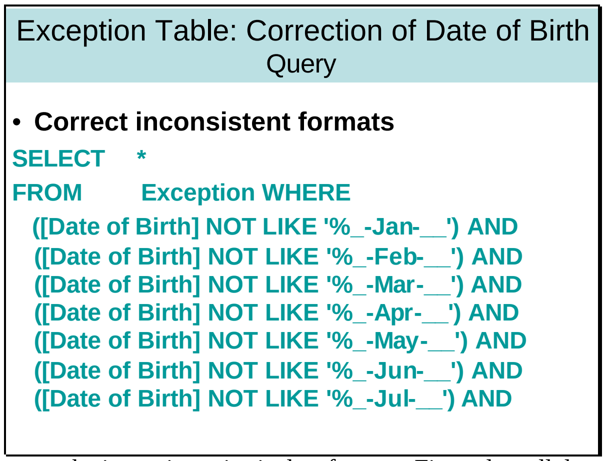
Exception Table: Correct Gender
Fill Rows Manually

In For Lahore campus we have only 187 missing values so we can fill 187 rows manually just by inspection of names.

- **How about same values (or more) for other three campus?**
- **Cant fill hundreds of gender values by hand!**

In this particular case of Lahore campus we have only 187 missing values so we can fill 187 rows manually just by inspection of names.

Slide 7



Exception Table: Correction of Date of Birth Query

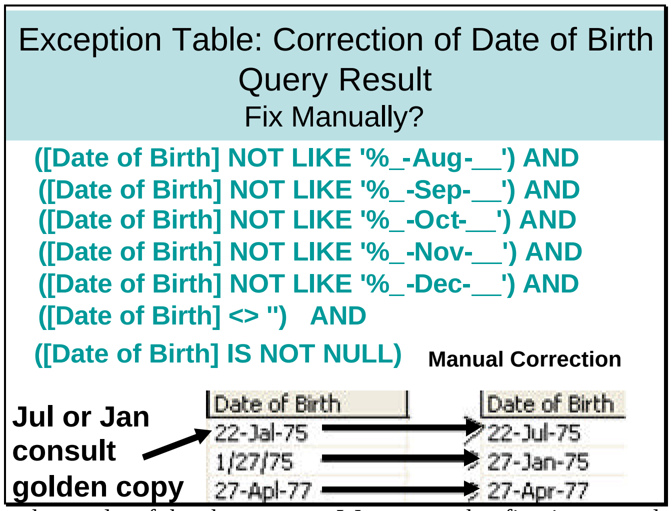
- **Correct inconsistent formats**

```
SELECT    *
FROM       Exception WHERE
  ([Date of Birth] NOT LIKE '%_-Jan-__') AND
  ([Date of Birth] NOT LIKE '%_-Feb-__') AND
  ([Date of Birth] NOT LIKE '%_-Mar-__') AND
  ([Date of Birth] NOT LIKE '%_-Apr-__') AND
  ([Date of Birth] NOT LIKE '%_-May-__') AND
  ([Date of Birth] NOT LIKE '%_-Jun-__') AND
  ([Date of Birth] NOT LIKE '%_-Jul-__') AND
```

Now we need to correct the inconsistencies in date formats. First select all dates with inconsistent formats. The following query can do this for us

```
SELECT    *
FROM      Exception WHERE
   ([Date of Birth] NOT LIKE '%_-Jan-__') AND
   ([Date of Birth] NOT LIKE '%_-Feb-__') AND
   ([Date of Birth] NOT LIKE '%_-Mar-__') AND
   ([Date of Birth] NOT LIKE '%_-Apr-__') AND
   ([Date of Birth] NOT LIKE '%_-May-__') AND
   ([Date of Birth] NOT LIKE '%_-Jun-__')  AND
   ([Date of Birth] NOT LIKE '%_-Jul -__')   AND
   ([Date of Birth] NOT LIKE '%_-Aug -__') AND
   ([Date of Birth] NOT LIKE '%_-Sep-__') AND
   ([Date of Birth] NOT LIKE '%_-Oct-__')  AND
   ([Date of Birth] NOT LIKE '%_-Nov-__') AND
   ([Date of Birth] NOT LIKE '%_-Dec-__') AND
   ([Date of Birth] <> '')   AND
   ([Date of Birth] IS NOT NULL)
```

Slide 8



Exception Table: Correction of Date of Birth Query Result
Fix Manually?

([Date of Birth] NOT LIKE '%_-Aug-__') AND
([Date of Birth] NOT LIKE '%_-Sep-__') AND
([Date of Birth] NOT LIKE '%_-Oct-__') AND
([Date of Birth] NOT LIKE '%_-Nov-__') AND
([Date of Birth] NOT LIKE '%_-Dec-__') AND
([Date of Birth] <> '')   AND
([Date of Birth] IS NOT NULL)

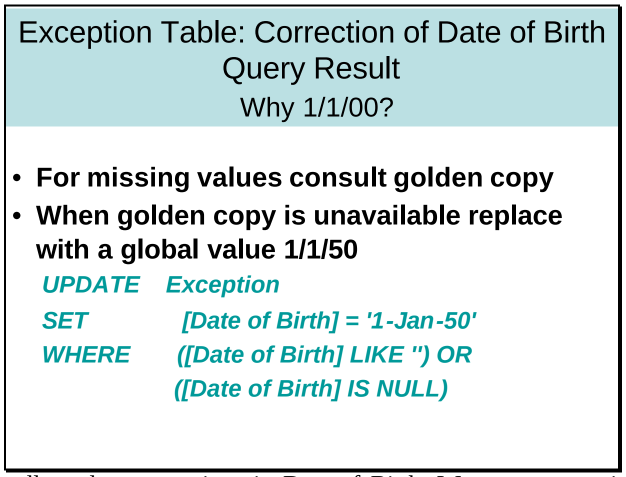
Manual Correction

Jul or Jan consult golden copy

Date of Birth	Date of Birth
22-Jal-75	22-Jul-75
1/27/75	27-Jan-75
27-Apl-77	27-Apr-77

The slide shows the results of the above query. We can see that first incorrect date is 22-Jal-75. This error can not be corrected until and unless we do not consult golden copy. Remaining two errors can be corrected without consulting golden copy.

Slide 9



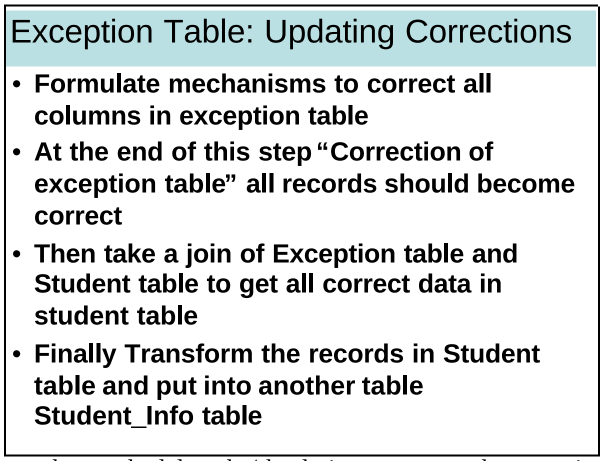
Exception Table: Correction of Date of Birth Query Result
Why 1/1/00?

- **For missing values consult golden copy**
- **When golden copy is unavailable replace with a global value 1/1/50**

```
UPDATE   Exception
SET         [Date of Birth] = '1-Jan-50'
WHERE      ([Date of Birth] LIKE '') OR
           ([Date of Birth] IS NULL)
```

There are some nulls and empty strings in Date of Birth. We can not put in values without consulting golden copy. So for the time being we can use a standard value like 1/1/1900 or 1/1/1950 etc., again a meta data entry. Later on these values can be replaced with original values after consulting golden copy.

Slide 10

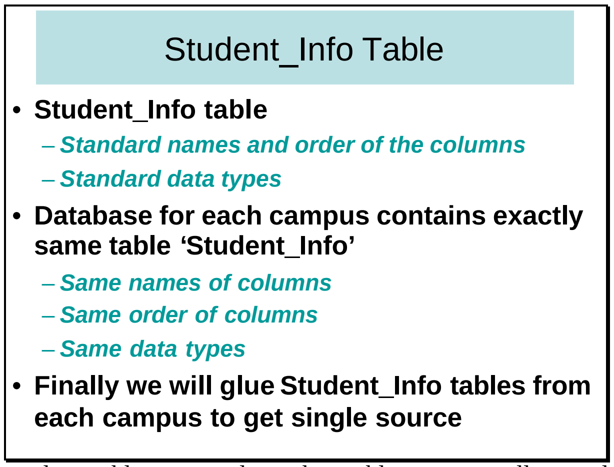


Exception Table: Updating Corrections

- **Formulate mechanisms to correct all columns in exception table**
- **At the end of this step "Correction of exception table" all records should become correct**
- **Then take a join of Exception table and Student table to get all correct data in student table**
- **Finally Transform the records in Student table and put into another table Student_Info table**

Its up to the designer what methodology he/she designs to correct the exception table. The goal is to correct all records in exception table up to a certain level. Hundred percent correction is not possible to achieve because of errors in golden copy. After correction of exception table we need to take a join of exception table and student table so that all records of student table can be updated with the values in exception table. After this exercise student table should be clean up to a variable level.
After the cleansing of all records it is the time to transform records and put them into another table named as 'Student_Info'.

Slide 11



Student_Info Table

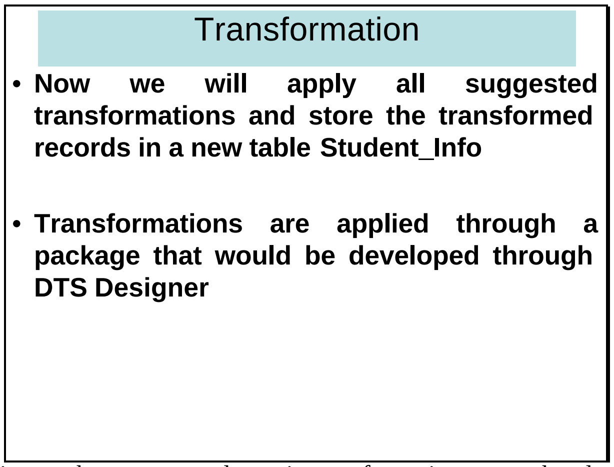
- **Student_Info table**
  - *Standard names and order of the columns*
  - *Standard data types*
- **Database for each campus contains exactly same table 'Student_Info'**
  - *Same names of columns*
  - *Same order of columns*
  - *Same data types*
- **Finally we will glue Student_Info tables from each campus to get single source**

After correcting student table we need another table to store all records after applying transformations. To serve this purpose we create another table Student_Info. The most important thing about this table is that it contains the names and data types of columns that are suggested in data profiling. For example, data type for dates columns (birth date and registration date) is DateTime, therefore, in Student_Info table we set data types for these columns dateTime at the time of creation. Same is the case with all other columns.
Student_Info table does not contain any column against row_id as this column was just added for cleansing purposes. Another important point is that Student_Info table may contain more or less

columns as compared to Student table. Like in Peshawar campus we do not have the column Gender but Student_Info table for Peshawar campus does contain it. This column can be filled by joining with the table Gender_guide we created earlier to find the missing genders.

Same query of create table is used to create table Student_Info in the databases of each campus, so that exactly the same table can be creat ed in all databases, with same names and same order of columns. After transformations and completion of Student_Info table we will just glue four tables (student_Info table, one from each campus) to get a single standardized table. As we know that in each campus order of columns in student table is different whereas order of columns in Student_Info table across each campus is same. Its mean with in a database, like Islamabad_Campus, order of columns in both tables Student and Student_Info differ. We need to apply 'copy column transformation' so that this different order of columns would not create any problem at the time of loading Student_Info table.

Slide 12

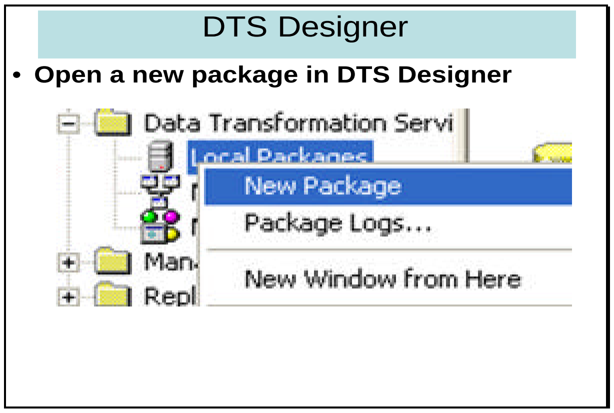


Transformation

- **Now we will apply all suggested transformations and store the transformed records in a new table Student_Info**
- **Transformations are applied through a package that would be developed through DTS Designer**

While data profiling we have suggested certain transformations on each column. Now it is the time to apply all transformations on Student table and finally put all transformed records to student_info table. To apply transformations we need to develop a package though DTS designer because wizard can not provide us enough functionality to design a package with complex transformations.

Slide 13

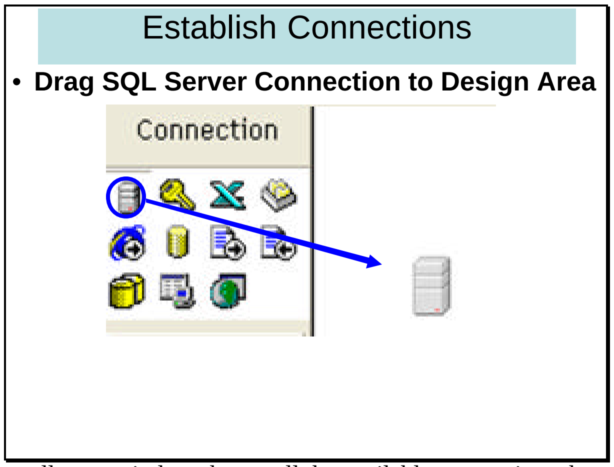


DTS Designer

- **Open a new package in DTS Designer**

To open a new package in DTS Designer. Right click the local packages and select ‘New Package’. As a result DTS Designer interface would open.

Slide 14

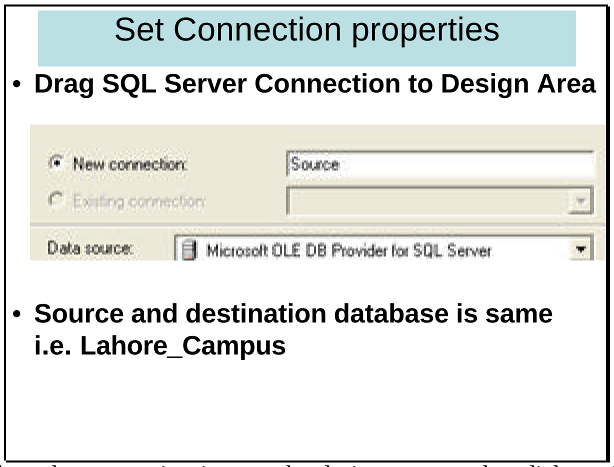


Establish Connections

- **Drag SQL Server Connection to Design Area**

On the left side a small pane window shows all the available connections that can be established through SQL Server. At this stage our source is Student table in Lahore_Campus SQL database. Therefore, for source connection we click at the SQL Server icon and drag it to the desing area.

Slide 15



Set Connection properties

- **Drag SQL Server Connection to Design Area**

[✓ New connection:] Source

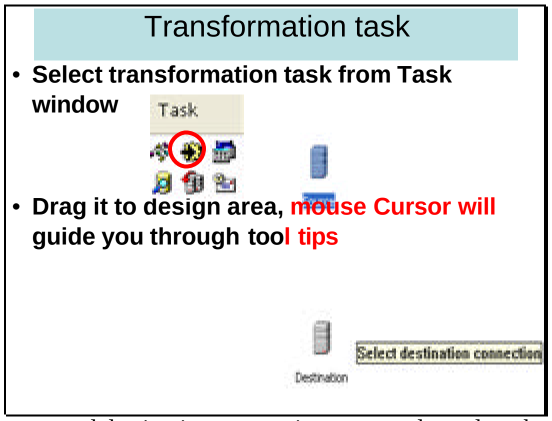
Existing connection:

Data source: Microsoft OLE DB Provider for SQL Server

- **Source and destination database is same i.e. Lahore_Campus**

As soon as we place the connection icon to the design area another dialog window opens and provides us the way to set the properties of theconnection. First of all it asks us whether we want to create a new connection or we want to use any other connection that was created before in the same package. If we want to use any existing connection we can select existing connection but here we want to create a new connection. Name the new connection as Source. Similar to source we need to create another SQL Server connection for destination table that is ‘Student_Info’. In properties of connection we are required to specify the name of destination database as well.

Slide 16

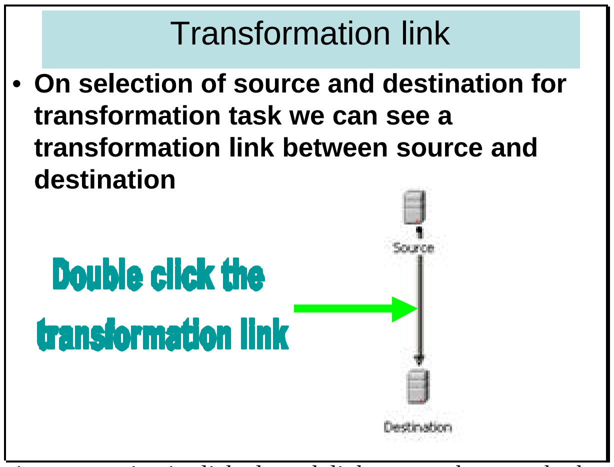


Transformation task

- **Select transformation task from Task window**
- **Drag it to design area, mouse Cursor will guide you through tool tips**

After creating both source and destination connections we need to select the task to be performed between source and destination. All available task can be found on the left pane window 'Task' in form of icons. The task is transformation and is represented by the icon we can see highlighted in the slide. Click the transformation task icon and drag it to the design area. First click in the source connection and <u>then</u> click on the <u>destination</u> connection.

Slide 17

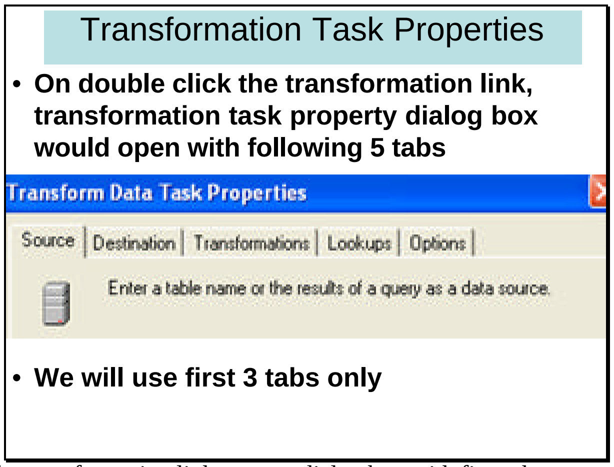


Transformation link

- **On selection of source and destination for transformation task we can see a transformation link between source and destination**

As soon as destination connection is clicked a task link appears between both connections. To set the properties and details of transformations (detail means what transformation needs to be applied) we need to double click the transformation link.

Slide 18



Transformation Task Properties

- On double click the transformation link, transformation task property dialog box would open with following 5 tabs

Transform Data Task Properties

Source | Destination | Transformations | Lookups | Options

Enter a table name or the results of a query as a data source.

- We will use first 3 tabs only

Double clicking the transformation link opens a dialog box with five tabs.
Source

Source tab let us define the source table.

Destination

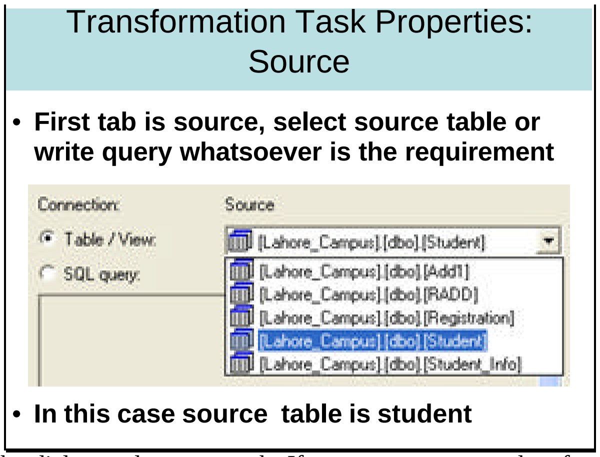
Destination tab let us define the destination table. In this case both will be the same.

Transformation

Transformation tab let us define the type of transformations to be applied between source and destination.

Rest of the two tabs **Lookup** and **Options** will not be used by us. As all the task required to be done can be completed by these three tabs.

Slide 19



Transformation Task Properties: Source

- First tab is source, select source table or write query whatsoever is the requirement

Connection: Source

Table / View: [Lahore_Campus].[dbo].[Student]

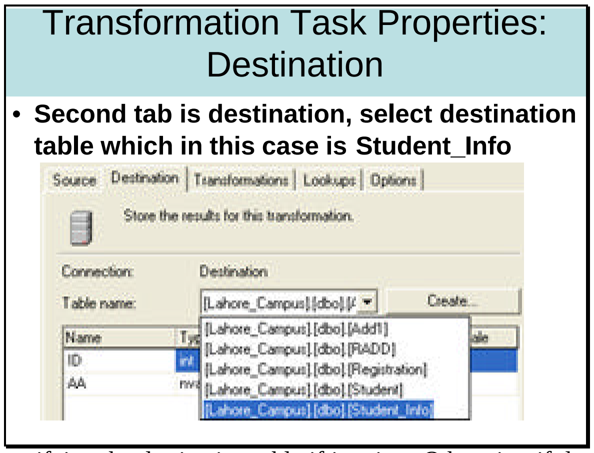
SQL query:

[Lahore_Campus].[dbo].[Add1]
[Lahore_Campus].[dbo].[RADD]
[Lahore_Campus].[dbo].[Registration]
[Lahore_Campus].[dbo].[Student]
[Lahore_Campus].[dbo].[Student_Info]

- In this case source table is student

The slide shows the dialog under source tab. If we want to extract data from source database through some query then we may specify query by selecting the radio option ‘SQL Query’. On the other hand, if we want to copy all columns of a table or a view then we may specify it through drop down menu after selecting the first radio option Table/View. The drop down menu shows all tables and views available in the database specified while setting the properties of the connection. In this case source table is student.

Slide 20



Transformation Task Properties: Destination

- **Second tab is destination, select destination table which in this case is Student_Info**

Store the results for this transformation.

Connection: Destination

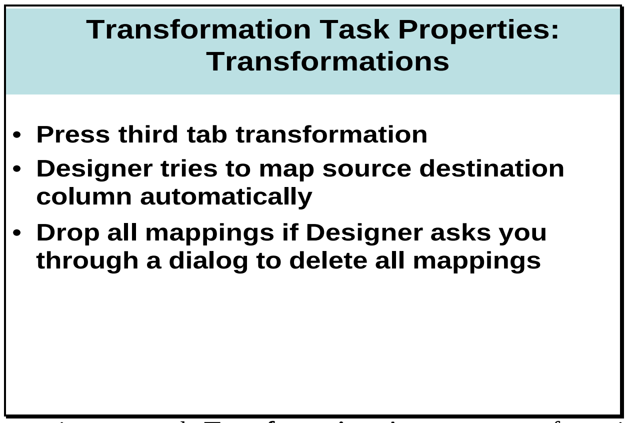
Table name: [Lahore_Campus].[dbo].[/

Name	Typ
ID	int
AA	nva

[Lahore_Campus].[dbo].[Add1]
[Lahore_Campus].[dbo].[RADD]
[Lahore_Campus].[dbo].[Registration]
[Lahore_Campus].[dbo].[Student]
[Lahore_Campus].[dbo].[Student_Info]

Second tab is for specifying the destination table if it exists. Otherwise, if destination table does not exist then we may create it through **Create** button on the right of drop down menu. In our case destination table Student_Info has been created earlier, therefore, at this stage we just need to select the Student_Info table from the drop down menu.

Slide 21



Transformation Task Properties: Transformations

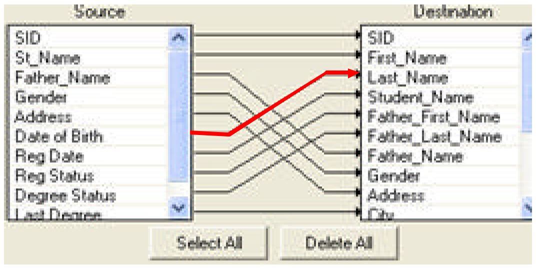
- **Press third tab transformation**
- **Designer tries to map source destination column automatically**
- **Drop all mappings if Designer asks you through a dialog to delete all mappings**

Now comes the most important tab **Transformation.** As soon as transformation tab is clicked designer tries to map the source and destination table columns automatically on the base of resemblances in the names. For example, if we have a column SID in source and a column SID in destination then designer tries to map these two columns. A dialog box appears and ask weather we want optimizer mappings remained there or we want to drop all mappings and create these mappings manually. We prefer to go for later option and drop all automatic mappings because optimizer does not transform genders (from 0/1 to M/F) or names etc.

Slide 22

Transformation Properties: Transformation Mappings

- **Delete all mappings**

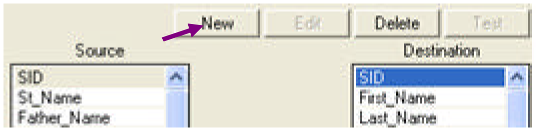


The slides show the mappings done by designer itself, delete all of these mappings.

Slide 23

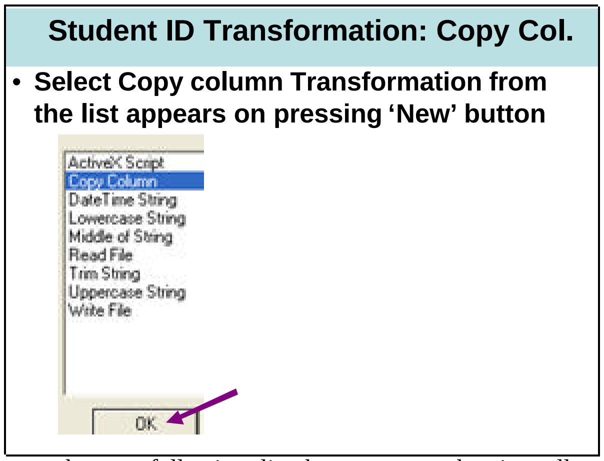
Student ID Transformation

- **Delete all auto mappings**
- **Select SID from both source and destination and Click new**



To create new mappings first of all we need to select the source column from source list box and then destination column from destination list box. When both gets highlighted as shown in the slide then press the **New** button. The slide shows highlighted SID column from both Source and destination list box. In both tables the first column is SID.

Slide 24

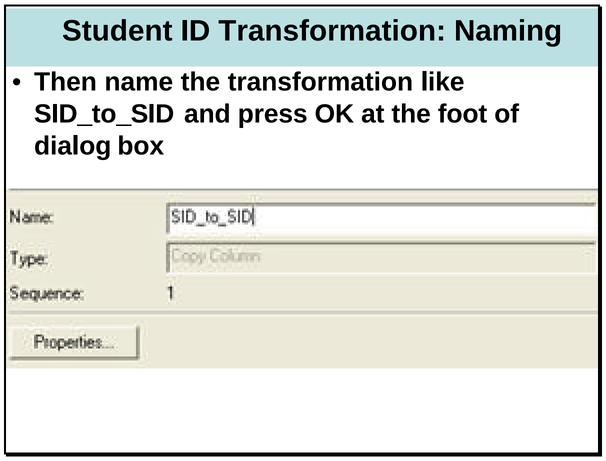


Student ID Transformation: Copy Col.

- Select Copy column Transformation from the list appears on pressing ‘New’ button

On pressing the new button, following list box appears showing all available forms of transformations. Select **Copy Column** transformation and press OK.

Slide 25



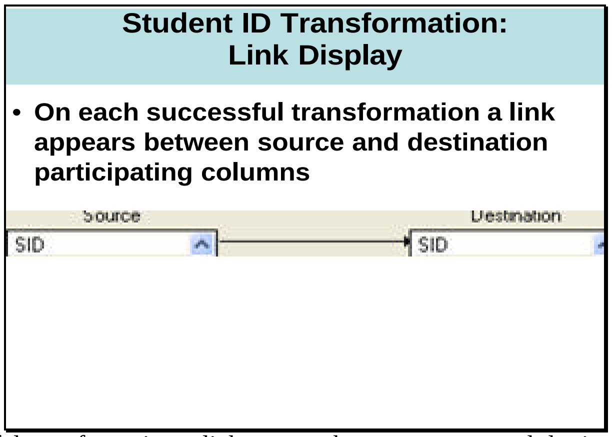
Student ID Transformation: Naming

- Then name the transformation like SID_to_SID and press OK at the foot of dialog box

Name: SID_to_SID
Type: Copy Column
Sequence: 1

After selecting the type of transformation needs to be applied we are required to assign name to this transformation like SID_to_SID. For this transformation we are not required to set the properties as it is the simplest form of transformation just copy from source to destination. The only difference is the size of variable. At source it was varchar[250] whereas at destination it is varchar[10]. Such transformations are done automatically by SQL Server, does not need to specify in properties.

Slide 26



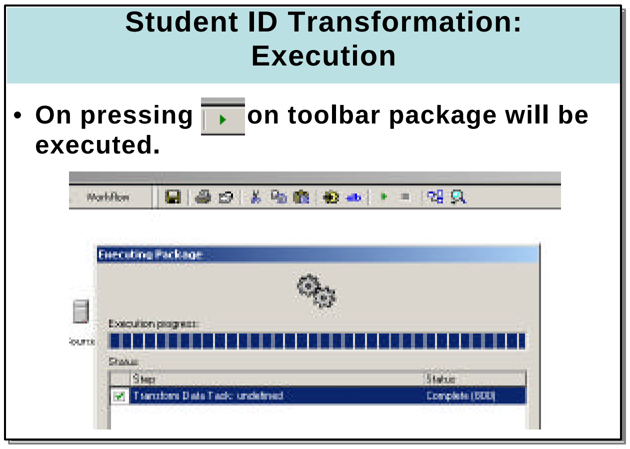
Student ID Transformation: Link Display

- On each successful transformation a link appears between source and destination participating columns

Source
Destination
SID
SID

On each successful transformation a link appears between source and destination participating columns. To modify the properties of transformation we are just need to double click the link and the same properties dialog box would appear.

Slide 27



Student ID Transformation: Execution

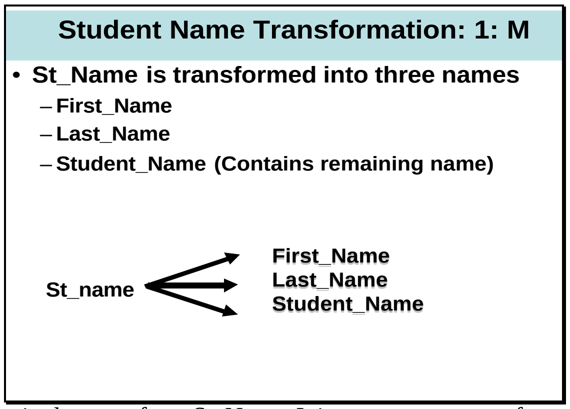
- On pressing on toolbar package will be executed.

Workflow
Executing Package
Execution progress:
Status:

Step	Status
Transform Data Task: undefined	Complete (000)

On each successful transformation a link appears between source and destination participating columns. To modify the properties of transformation we are just need to double click the link and the same properties dialog box would appear. .

Slide 28



Student Name Transformation: 1: M

- **St_Name is transformed into three names**
  - **First_Name**
  - **Last_Name**
  - **Student_Name (Contains remaining name)**

$$\begin{tikzcd} & \text{First\_Name} \\ \text{St\_name} \arrow[ur] \arrow[r] \arrow[dr] & \text{Last\_Name} \\ & \text{Student\_Name} \end{tikzcd}$$

After SID we are required to transform St_Name. It is one to many transformation that is required to be specified through script. In this case is source column is just one 'St_Name' and the destination columns are three 'First_Name', 'Last_Name', & 'Student_Name'. We are required to create three new transformations,

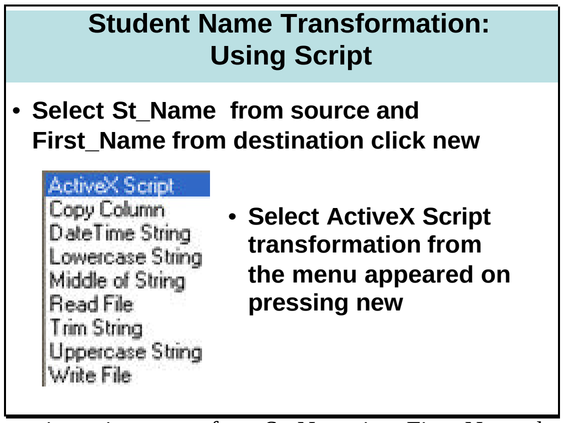
St_Name to First_Name

St_Name to Last_Name

St_Name to Student_Name

First of all select St_Name from source list box and First_Name from destination list box. When both of the columns get highlighted press the new button.

Slide 29



Student Name Transformation: Using Script

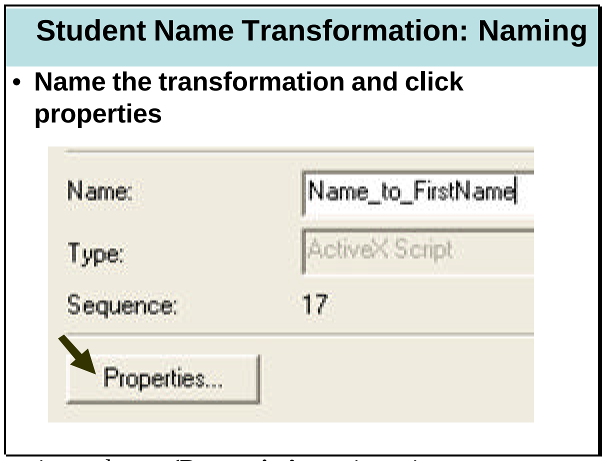
- **Select St_Name from source and First_Name from destination click new**

ActiveX Script
Copy Column
DateTime String
Lowercase String
Middle of String
Read File
Trim String
Uppercase String
Write File

- **Select ActiveX Script transformation from the menu appeared on pressing new**

As we are required to write script to transform St_Name into First_Name therefore select Active-X Script.

Slide 30



Student Name Transformation: Naming

- Name the transformation and click properties

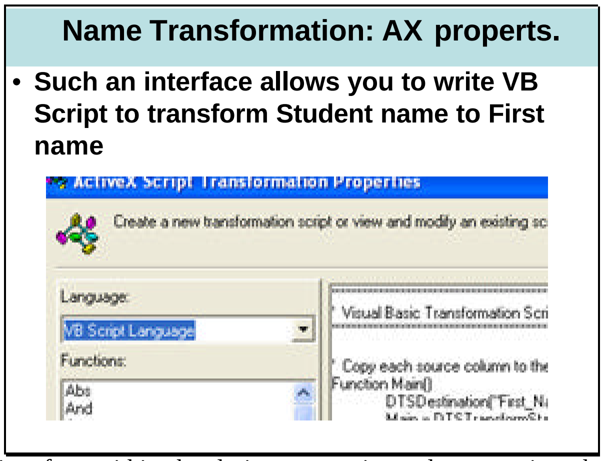
Name: Name_to_FirstName

Type: ActiveX Script

Sequence: 17

Name the transformation and press **'Properties'** to write script.

Slide 31



Name Transformation: AX properts.

- Such an interface allows you to write VB Script to transform Student name to First name

ActiveX Script Transformation Properties

Create a new transformation script or view and modify an existing sc

Language: VB Script Language

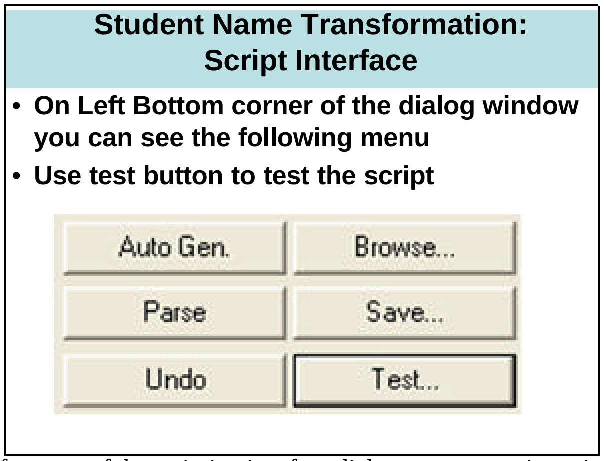
Functions:

Abs
And

```
' Visual Basic Transformation Scri

' Copy each source column to the
Function Main()
        DTSDestination("First_N
```

Slide shows the interface within the designer to write and test scripts that are written for transformation. Language dropdown menu on the extreme left corner allow us to select options available for scripting the transformation. As we are using VB scripts therefore make sure that VB Script should be selected in the dropdown menu.

Slide 32

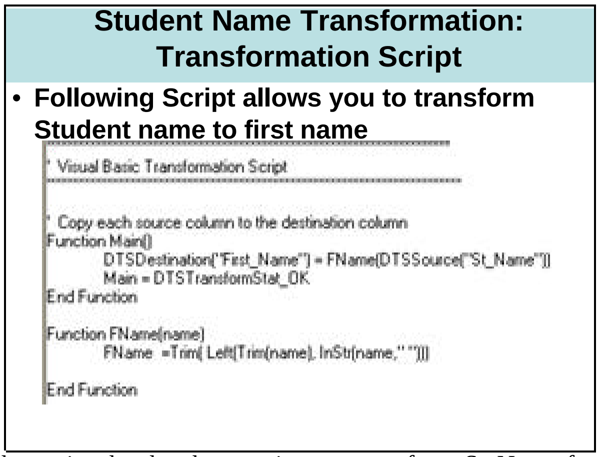


Student Name Transformation: Script Interface

- **On Left Bottom corner of the dialog window you can see the following menu**
- **Use test button to test the script**

On the extreme left corner of the scripting interface dialog we can see six options as shown in the slide. We can save our scripts, we can browse them later on, we can parse them for syntax errors, we can undo last move, we can auto generate simple copy column script and we can test our scripts on actual column of database table. To verify the correctness of the script we can use **Test** option. Test option runs the script on the actual column and shows the first hundred values as an output.

Slide 33



Student Name Transformation: Transformation Script

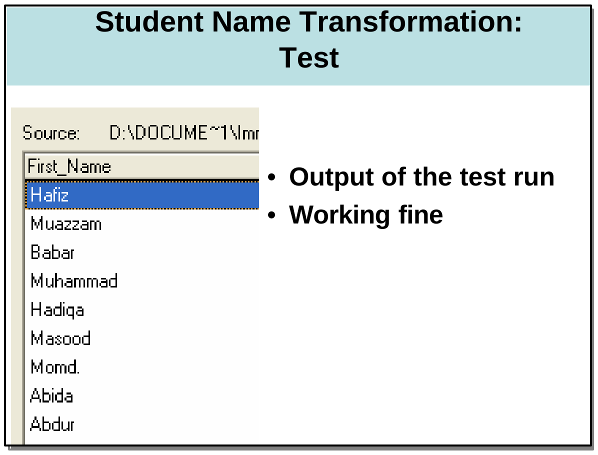
- **Following Script allows you to transform Student name to first name**

```
'**********************************************************************
'  Visual Basic Transformation Script
'************************************************************************


'  Copy each source column to the destination column
Function Main()
        DTSDestination("First_Name") = FName(DTSSource("St_Name"))
        Main = DTSTransformStat_OK
End Function

Function FName(name)
        FName  =Trim( Left(Trim(name), InStr(name," ")))

End Function
```

The slide shows the script that has been written to transform St_Name from source table to First_Name of destination. If you do not know VB Script you can also use same script with correct column names.

Slide 34

Slide 80



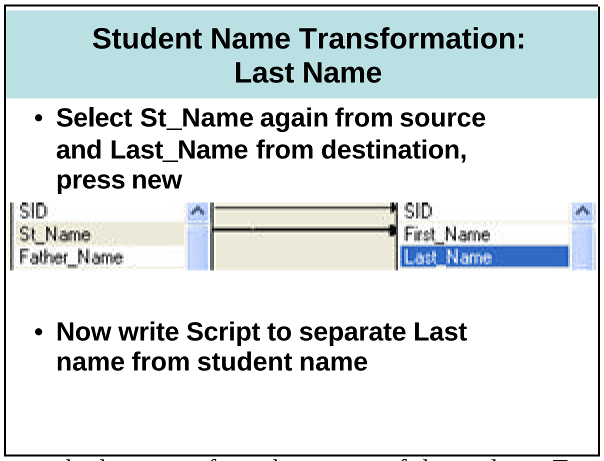
Student Name Transformation: Test

```
Source:    D:\DOCUME~1\Imr
First_Name
Hafiz
Muazzam
Babar
Muhammad
Hadiqa
Masood
Momd.
Abida
Abdur
```

- **Output of the test run**
- **Working fine**

The slide shows the output of the test run of given script on Lahore_Campus database. We can see all first names are separated from the actual full names. This is the desired transformation for First_Name.

Slide 35



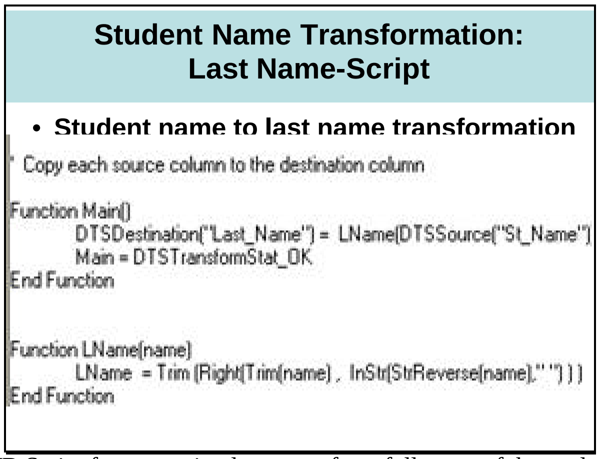
Student Name Transformation: Last Name

- **Select St_Name again from source and Last_Name from destination, press new**

- **Now write Script to separate Last name from student name**

Now we will separate the last name from the names of the students. To create this type of transformation we are required to selectSt_Name from source list box and **Last_Name** from the destination list box and press **New button**.

Slide 36



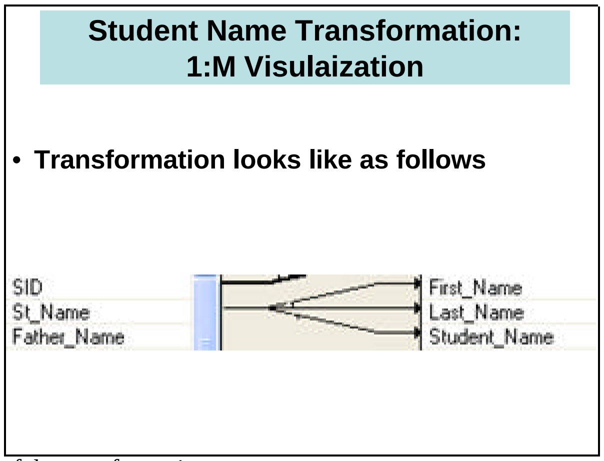
Student Name Transformation: Last Name-Script

- Student name to last name transformation

```
' Copy each source column to the destination column

Function Main()
        DTSDestination("Last_Name") = LName(DTSSource("St_Name"))
        Main = DTSTransformStat_OK
End Function


Function LName(name)
        LName  = Trim (Right(Trim(name) ,  InStr(StrReverse(name)," ") ) )
End Function
```

The slide shows VB Script for separating last name from full name of the student.

Slide 37

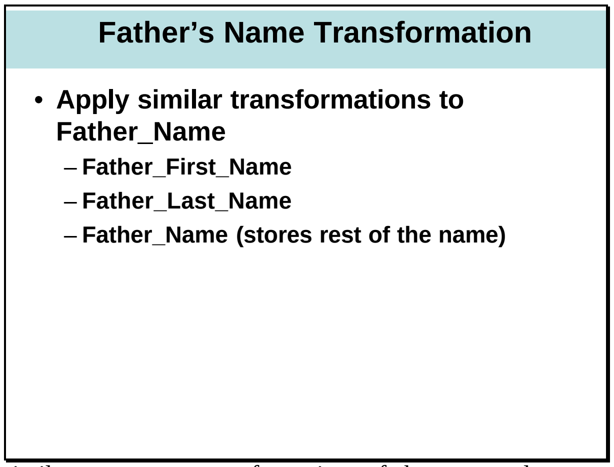


Student Name Transformation: 1:M Visulaization

- Transformation looks like as follows

After creating all of the transformations
St_Name to First_Name
St_Name to Last_Name
St_Name to Student_Name
Link for one to many transformations looks like the one shown in the slide.

Slide 38

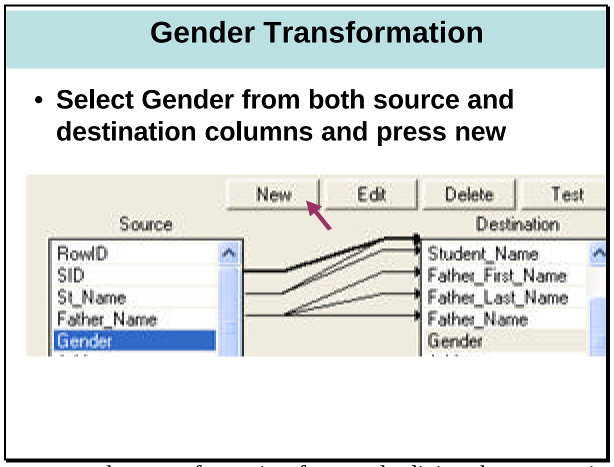


Father's Name Transformation

- Apply similar transformations to Father_Name
  - Father_First_Name
  - Father_Last_Name
  - Father_Name (stores rest of the name)

We need to apply similar one to many transformation to father name column.

Slide 39

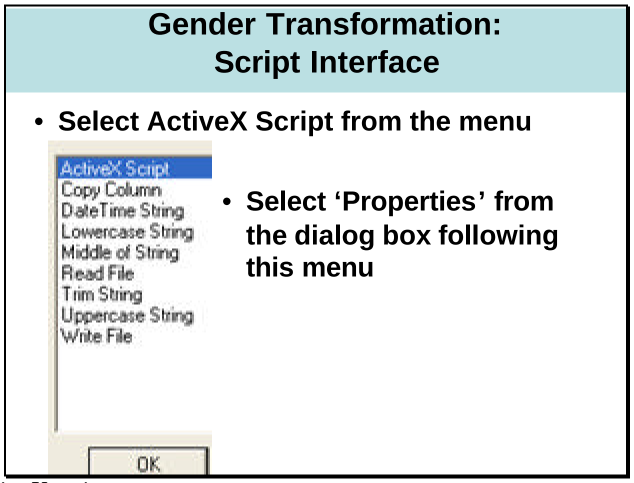


Gender Transformation

- Select Gender from both source and destination columns and press new

Now we need to create another transformation for standardizing the conventions to store gender that is **M** for male and **F** for female. Select Gender from both source and destination and press **'New'.**

Slide 40

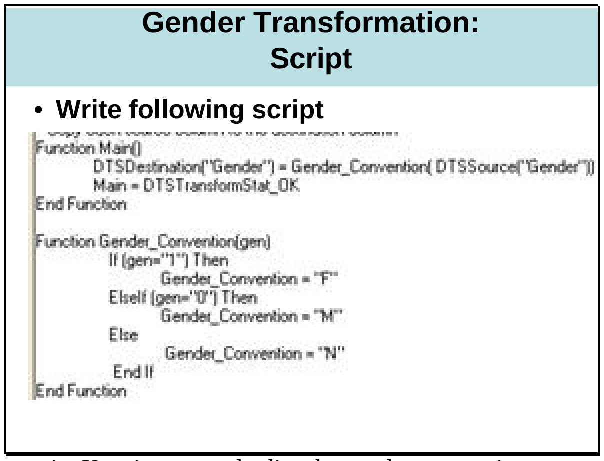


Gender Transformation: Script Interface

- Select ActiveX Script from the menu
- Select ‘Properties’ from the dialog box following this menu

Again select ActiveX script.

Slide 41



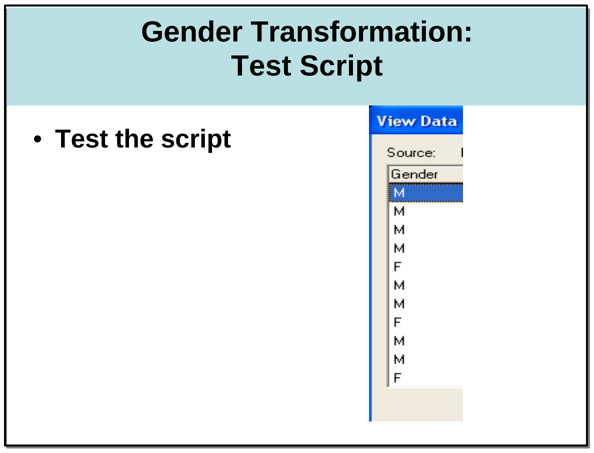
Gender Transformation: Script

- Write following script

```
Function Main()
        DTSDestination("Gender") = Gender_Convention( DTSSource("Gender"))
        Main = DTSTransformStat_OK
End Function

Function Gender_Convention(gen)
          If (gen="1") Then
                    Gender_Convention = "F"
          ElseIf (gen="0") Then
                    Gender_Convention = "M"
          Else
                     Gender_Convention = "N"
           End If
End Function
```

The slide shows the activeX script to standardize the gender convention.

Slide 42



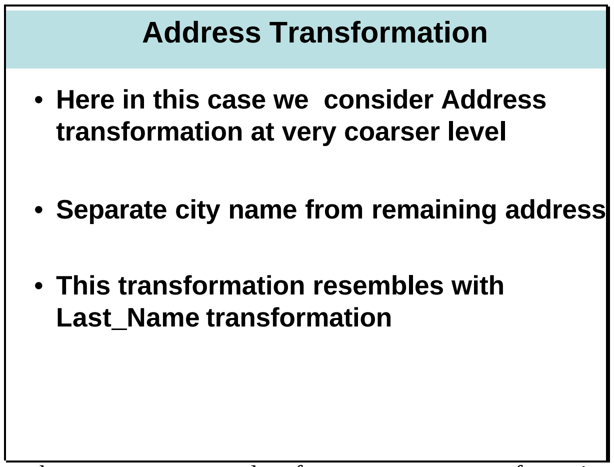
Gender Transformation: Test Script

- Test the script

Gender
M
M
M
M
F
M
M
F
M
M
F

Test run shows that the script is working fine.

Slide 43

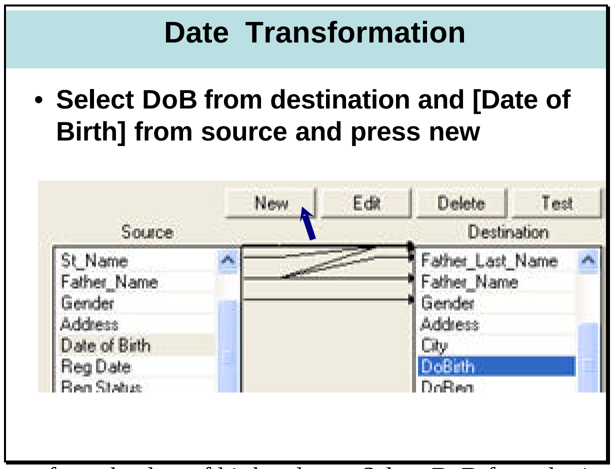


Address Transformation

- Here in this case we consider Address transformation at very coarser level
- Separate city name from remaining address
- This transformation resembles with Last_Name transformation

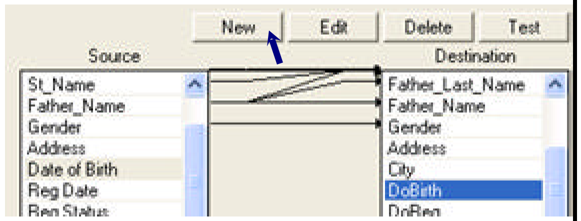
Now comes to another common example of one to many transformation that is Address transformations. For real life purposes we usuallly break address into many columns like house #, Street #, City, State, Region etc. It depends on the requirements of the users and customers. For the given example we will break the address into two column City and remaining address. This example resembles to the separation of last name from student full name because in all records city is the last part of the address. So we can use the same script with certain modification of column names.

Slide 44



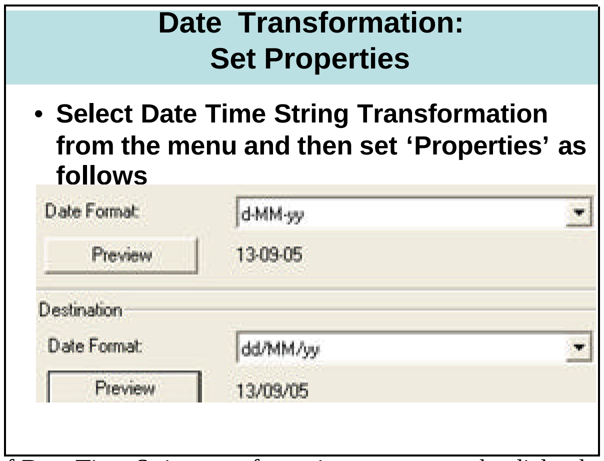
Date Transformation

- Select DoB from destination and [Date of Birth] from source and press new



Now we need to transform the date of birth column. Select **DoB** from destination and **[Date of Birth]** from source and press **New**. Select **Date Time String** transformation from the list of available transformations.

Slide 45



Date Transformation: Set Properties

- Select Date Time String Transformation from the menu and then set ‘Properties’ as follows

Date Format: d-MM-yy

Preview 13-09-05

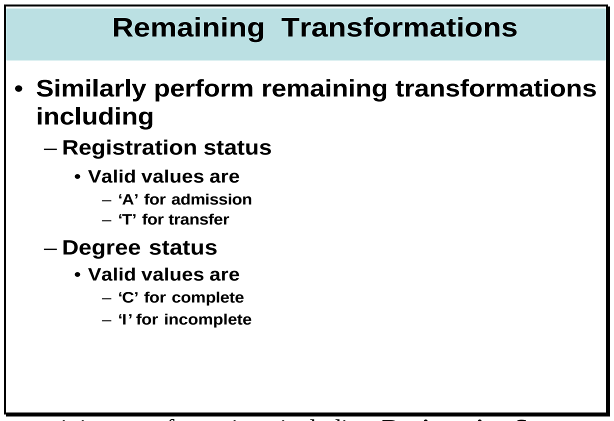
Destination

Date Format: dd/MM/yy

Preview 13/09/05

In the properties of Date Time String transformation we can see the dialog box as shown in the slide. In this dialog box we can see two dropdown combo boxes with title **Date Format**. In first combo box select or manually specify (through typing) the format of the date in the source column. In the second box select the format for destination and click **ok**.

Slide 46

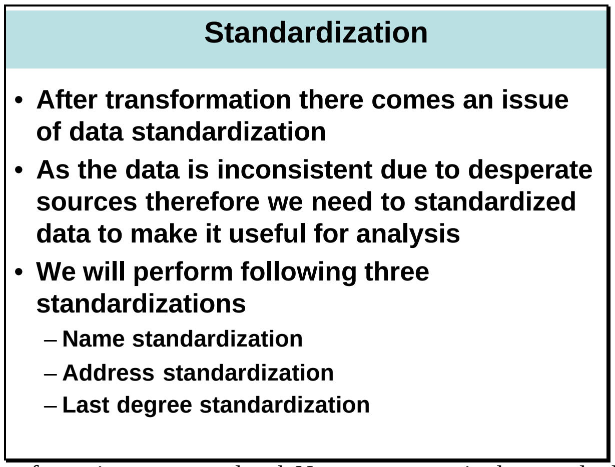


Remaining Transformations

- Similarly perform remaining transformations including
  - Registration status
    - Valid values are
      - ‘A’ for admission
      - ‘T’ for transfer
  - Degree status
    - Valid values are
      - ‘C’ for complete
      - ‘I’ for incomplete

Similarly perform remaining transformations including **Registration Status** and **Degree Status**. You can generate scripts for them easily by performing little modifications in gender transformation script.

Slide 47

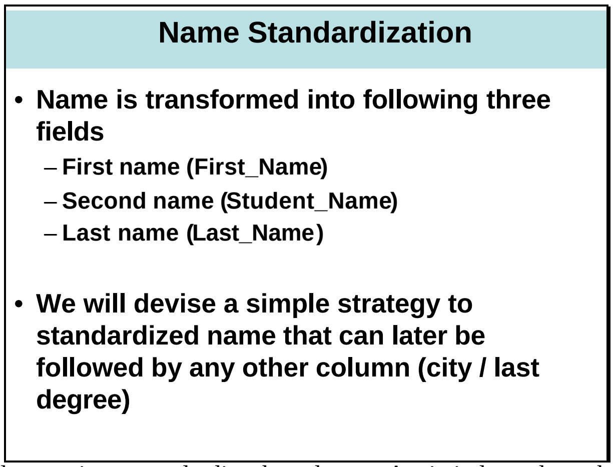


Standardization

- After transformation there comes an issue of data standardization
- As the data is inconsistent due to desperate sources therefore we need to standardized data to make it useful for analysis
- We will perform following three standardizations
  - – Name standardization
  - – Address standardization
  - – Last degree standardization

By this time all transformations are completed. Now we are required to standardize data. Like we can see in the name column that there are a lot of inconsistencies in names due to variations in spellings. We can find many variations in spellings of same name like Mohd., Mohammed, Muhammed, Mohamad, etc.. For meaning full analysis we must have standardized data in our columns because computer can not recognize that Mohd. Khalid and Mohammad Khalid is the same name. Same is the case with the names of cities. Some people use abbreviations in addresses while some others like to write full names e.g. Lahore or Lhr. In this example we will perform standardization for Names, Addresses and Last degree.

Slide 48

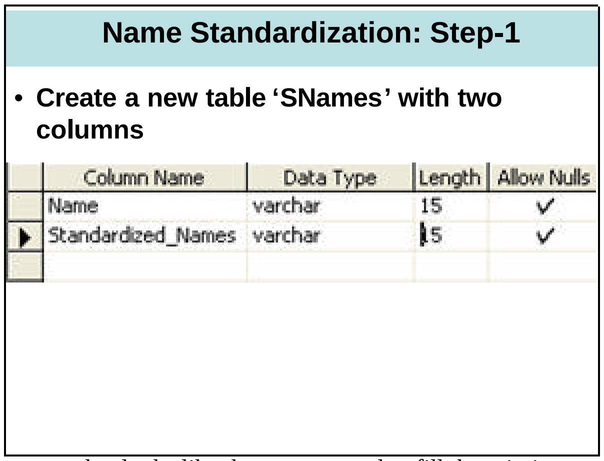


Name Standardization

- Name is transformed into following three fields
  - – First name (First_Name)
  - – Second name (Student_Name)
  - – Last name (Last_Name)
- We will devise a simple strategy to standardized name that can later be followed by any other column (city / last degree)

There are no fixed strategies to standardize the columns. Again it depends on the project designer what methodology he/she devises. We can devise a simple methodology that can later be used for other columns as well.

Slide 49



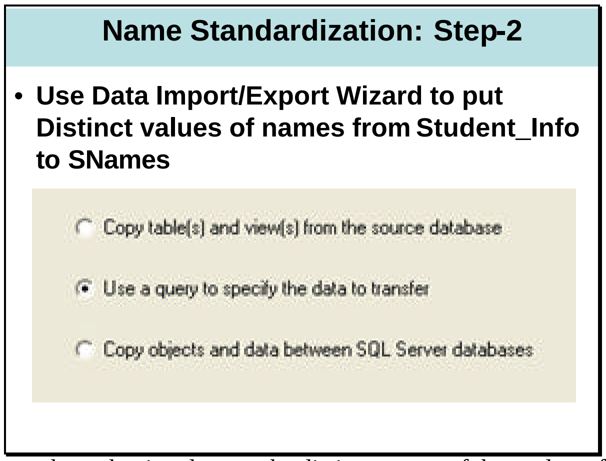
Name Standardization: Step-1

- **Create a new table 'SNames' with two columns**

	Column Name	Data Type	Length	Allow Nulls
	Name	varchar	15	✓
▶	Standardized_Names	varchar	15	✓

This methodology some what looks like the one we used to fill the missing gender information of students. Create a new table with two columns 'Name' and 'Standardized_Names'.

Slide 50



Name Standardization: Step-2

- **Use Data Import/Export Wizard to put Distinct values of names from Student_Info to SNames**

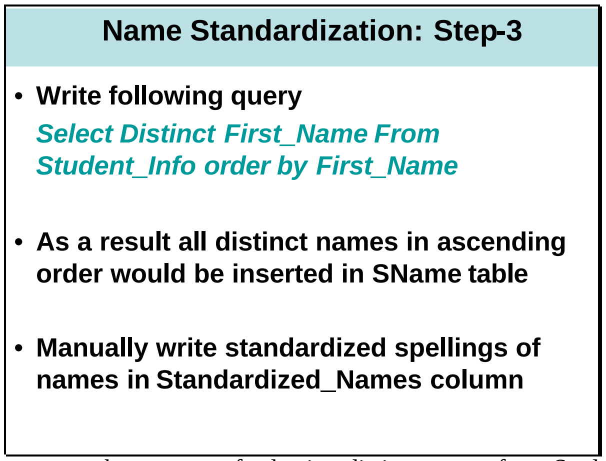
○ Copy table(s) and view(s) from the source database

[✓ Use a query to specify the data to transfer]

○ Copy objects and data between SQL Server databases

Create a new package through wizard to put the distinct names of the students from Student_Info table. Select second radio option that is use a query to specify the data to transfer. Write a query to select distinct names from Student_Info table.

Slide 51



Name Standardization: Step-3

- **Write following query**

  Select Distinct First_Name From Student_Info order by First_Name

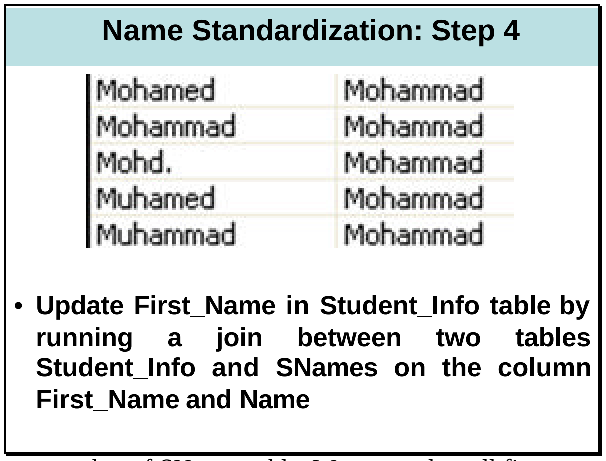
- **As a result all distinct names in ascending order would be inserted in SName table**

- **Manually write standardized spellings of names in Standardized_Names column**

Following query can serve the purpose of selecting distinct names from Student_Info table and loading them to SName table in ascending order.

Select Distinct First_Name From Student_Info order by First_Name;
Load all names to Name column of the SName table and against each name write its standardized spellings in Standardized_Names column manually.

Slide 52



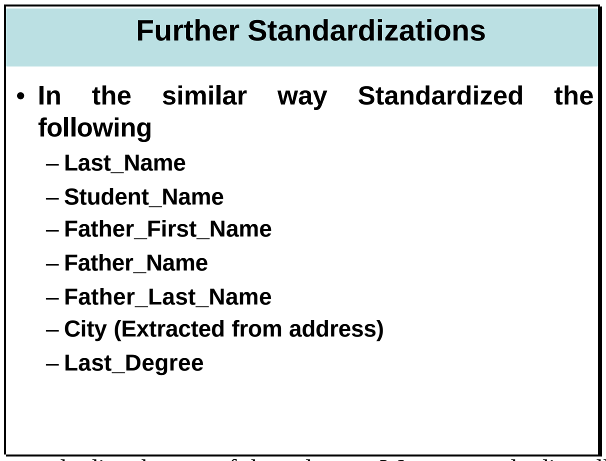
Name Standardization: Step 4

Mohamed	Mohammad
Mohammad	Mohammad
Mohd.	Mohammad
Muhamed	Mohammad
Muhammad	Mohammad

- **Update First_Name in Student_Info table by running a join between two tables Student_Info and SNames on the column First_Name and Name**

The slide shows a screen shot of SNames table. We can update all first names in Student_Info table just by joining it with SName on the column First_Name (Student_Info) and Name (SNames).

Slide 53



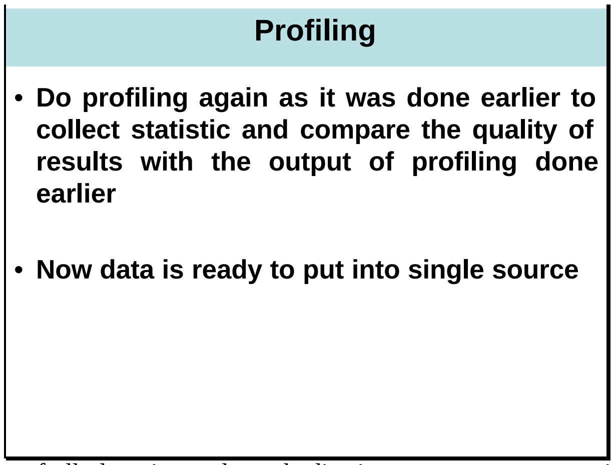
Further Standardizations

- In the similar way Standardized the following
  - Last_Name
  - Student_Name
  - Father_First_Name
  - Father_Name
  - Father_Last_Name
  - City (Extracted from address)
  - Last_Degree

In the similar way standardize the rest of the columns. We can standardize all of the following columns with the same methodology:

Last_Name
Student_Name
Father_First_Name
Father_Name
Father_Last_Name
City (Extracted from address)
Last_Degree

Slide 54



Profiling

- Do profiling again as it was done earlier to collect statistic and compare the quality of results with the output of profiling done earlier
- Now data is ready to put into single source

To test the quality of all cleansing and standardization process we are required to profile all columns again. This time there should be no missing and invalid values in the data. As we can not attain hundred percent cleanliness level therefore for remaining errors and invalidities in the data we should have the proper reasoning.
After all this exercise now we are ready to put all data into single source.