ISEE84 DATA WAREHOUSING AND MINING

UNIT – II Data Mining Introduction

<u>INTRODUCTION</u>

<u>Data Mining:</u>

An Expanding universe of data We live in an expanding universe of data in which there is too much data and too little information. The development of new techniques to find the required information from huge amounts of data is one of the main challenges for software developers today. The quantity of data in the world roughly doubles every year, and as a somewhat surprising consequence, the amount of meaningful information decreases rapidly. The very fact that the amount of information is growing is the reason it is increasingly difficult to find the meaningful facts we seek. We can't see the wood for the trees. For the most part, this growth of information is due to the mechanical production of texts. Data mining is also known as data discovery and knowledge discovery E.g.: Pharmacy Company, Credit Card Company, Transport Company etc...,

Evolution of Data Mining:

In 1960, Collection of Data In 1980, Data accessing

In 1990, Data Warehousing

Today, Data Mining

	8 bits	1 byte
1000	1000 bytes	1 kilobyte
10002	1000 000	1 megabyte
10003	1000 000 000	1 gigabyte
10004	1000 000 000 000	1 terabyte
10005	1000 000 000 000 000	1 petabyte
10006	1000 000 000 000 000 000	1 exabyte
10007	1000 000 000 000 000 000 000	1zettabyte
10008	1000 000 000 000 000 000 000 000	1 yottabyte
10009	1000 000 000 000 000 000 000 000 000	1 brontobyte
100010	1000 000 000 000 000 000 000 000 000 000	1 geopbyte

Types of Data

Data mining can be performed on following types of data

Relational databases

Data warehouses

Advanced DB and information repositories

Object-oriented and object-relational databases

Transactional and Spatial databases

Heterogeneous and legacy databases

Multimedia and streaming database

Text databases

Text mining and Web mining

TECHNIQUES FOR DATA MINING IN CRM

Anomaly Detection

Searching for information that doesn't match expected behavior or a projected pattern is called anomaly detection. Anomalies can provide actionable information because they deviate from the average in the data set.

Association Rule Learning

Discover relations between data items in huge databases. With Association Rule Learning, hidden patterns can be uncovered and the information gained may be used to better understand customers, learn their habits, and predict their decisions.

Clustering

Identify similar data sets and understand both the similarities and the differences within the data. Data sets that have similar traits can be used for conversion rate increases. For example, if the buying behavior of one group of customers is similar to that of another group, they can both be targeted with similar services or products.

Classification

This technique is used for gathering information about data so that the data sets can be placed into proper categories. One example is the classification of email as either regular, acceptable email or as spam.

Regression

Regression analysis is one of the advanced data mining techniques in CRM. The objective is to find the dependency between different data items and map out which variables are affected by other variables. This technique is used to determine customer satisfaction levels and its impact on customer loyalty.

Data mining and Ethics

1. Selecting the wrong problem for data mining
2. Ignoring, What your sponsor thinks data mining is, and what it can and cannot do
4. Leaving insufficient time for data preparation
5. Looking only at aggregated results, never at individual records.
6. Being nonchalant about keeping track of mining procedure and results
7. Ignoring suspicious findings in your haste to move on
8. Running mining algorithms repeatedly without thinking hard enough about the next Stages of the data analysis
9. Believing everything you are told about the data
10. Believing everything you are told about your own data mining analyses
11. Measuring your results differently from the way your own sponsor will measure them

Nuggets of Data Mining

· User friendly, intuitive interface
· Available for desktop or as on line real time data mining engine
· Power to extract knowledge from data that other methods can not
· Automatic rule generation in English "if-then" rules
· Ability to model up to 50,000 variables (without using clustering)
· Employs machine learning (No statistics used)
· Automatic binning of numeric variables
· Binning of nominal variables
· Ability to handle complex non-linear relationships with no statistical requirements
· Handles missing data
· Handles noisy data
· Assists in finding data errors
· Provides validation module
· Provides predictions for new data
· Reverse engineers information implicit in databases

Data Mining Technologies

· Allows stratified sampling for training files

· Unique feature that resolves rule conflicts for better predictions

· Computes attribute significance without limitation of correlated variables

Areas of Potential Application

The following list includes only a few of the possible applications.

Business

· CRM

· Banking -- mortgage approval, loan underwriting, fraud analysis and detection

· Finance -- analysis and forecasting of business performance, stock and bond analysis

· Insurance -- bankruptcy prediction, risk analysis, credit and collection models

· Web Marketing – personalization, targeted banner ads and cross sell/upsell opportunities

· Direct Marketing – response models, churn models, optimum creative, next to buy analysis

· Government – threat assessment, terrorist profiling

Manufacturing

· Fault analysis, quality control, preventive maintenance scheduling, automated systems

Medicine

· Gene analysis, epidemiological studies, toxicology, diagnosis, drug interactions, risk factor analysis,

quality control, retrospective drug studies

Scientific Research

· General modeling of all types

INFORMATION AND PRODUCTION FACTOR

Most international organizations produce more information in a week than many people could read in a lifetime. The situation is even more alarming in worldwide networks like the internet. Every day hundreds of megabytes of data are distributed around the world. bur it is no longer possible to monitor this increasingly rapid development — the growth is exponential. We are confronted with the new paradox of the growth of data that more data means less information. In the future, the ability to read and interpret alone will not be enough to survive as a professional, a scientist or a commercial organization. The mechanical production and reproduction of data force us to adapt our strategies and develop mechanical methods for filtering. selecting. and interpreting data. Organizations that excel in doing this will have better chance of surviving. and because of this, information itself has become a production factor of importance. This tendency is perhaps most obvious in the stock exchange. where it is not only the availability of data that is vital but also the ability to interpret the data. and to act on the basis of these interpretations. The operation of the stock exchange has. to a large extent, become a game of computers against computers. with humans intervening only at a meta-level.

KDD AND DATA MINING

There is confusion about the exact meaning of the terms data mining' and `KDD' with many authors regarding them as synonymous. At the first international KDD conference in Montreal in 1995, it was proposed that the term `KDD' be employed to describe the whole process of extraction if knowledge from data. In this context, knowledge means relationships and patterns between data elements. It was further proposed that the term 'data mining' should be used exclusively for the discovery stage of the KDD process. In this book we will adopt these interpretations

A more or less official definition of KDD is: 'the non-trivial extraction of implicit, previously unknown and potentially useful knowledge from data.' So the knowledge must be new not obvious, and one must be able to use it. KDD is not a new technique but rather a multi-disciplinary field of research; machine learning, statistics, database technology, expert systems, and data visualization all make a contribution.

DATA MINING VS QUERY TOOLS

The main difference between data mining and query tools are

1. Firstly, it is important to realize that query tools and data mining tools are complementary.

2. A data mining tools does not replace a query tool, but it does give the user a lot of additional possibilities.

3. If you know exactly what you are looking for, use SQL, but if you know only vaguely what you we looking for, turn to data mining.

DATA MINING IN MARKETING

Although it is still a relatively new technology, businesses from all industry verticals i.e. healthcare, manufacturing, financial, transportation, etc. have invested in data mining technology to take advantage of historical data. Data mining techniques in CRM assist your business in finding and selecting the relevant information that can then be used to get a holistic view of the customer life-cycle; this comprises of four stages: customer identification, attraction, retention, and development. The more data there is in the database, the more accurate the models will be created and their subsequent use will result in more business value.

Data mining typically involves the use of predictive modeling, forecasting and descriptive modeling techniques as its key elements. Exploiting CRM in this age of data analytics enables an organization to manage customer retention, select the right prospects & customer segments, set optimal pricing policies, and objectively measure and rank the suppliers best suited for their needs.

Basket Analysis

Ascertain which items customers tend to purchase together. This knowledge can improve stocking, store layout strategies, and promotions.

Sales Forecasting

Examining time-based patterns helps businesses make re-stocking decisions. Furthermore, it helps you in supply chain management, financial management and gives complete control over internal operations.

Database Marketing

Retailers can design profiles of customers based on demographics, tastes, preferences, buying behavior, etc. It will also aid the marketing team in designing personalized marketing campaigns and promotional offers. This will result in enhanced productivity, optimal allocation of the company's resources and desirable ROI.

Predictive Life-Cycle Management

Data mining helps an organization predict each customer's lifetime value and to service each segment appropriately.

Market Segmentation

Learn which customers are interested in purchasing your products and design your marketing campaigns and promotions keeping their tastes and preferences in mind. This will increase efficiency and result in the desired ROI since you won't be targeting customers who show little to no interest in your product.

Product Customization

Manufacturers can customize products according to the exact needs of customers. In order to do this, they must be able to predict which features should be bundled to meet customer demand.

Fraud Detection

By analyzing past transactions that were later determined to be fraudulent, a business can take corrective measures and stop such events from occurring in the future. Banks and other financial institutions will benefit from this feature immensely, by reducing the number of bad debts.

Warranties

Manufacturers need to predict the number of customers who will make warranty claims and the average cost of those claims. This will ensure efficient and effective management of company funds.

PRACTICAL APPLICATIONS OF DATA MINING

• Lack of long-term vision: you need to ask 'what do we want to get from out files in the future?'

• Not all files are up to data: data is missing or incorrect; tiles vary greatly in quality.

• Struggle between departments: some departments do not want to give up their data.

• Poor cooperation from the electric data processing department: for example, 'they say just give us the queries and we will find the information you want'.

• Legal and privacy restrictions: some data cannot be used, for reasons of privacy.

• Files are hard to connect for technical reasons: there is a discrepancy between a hierarchical and a relational database, or data models are not up to date.

• Timing problems: files can be compiled centrally, but with a six month delay.

• Interpretation problems: connections are found in the database, but no one knows their meaning or what they can be used for.

LEARNING

Learning denotes changes in the system that, enables the system to do the same task more efficiently the next time. Learning is making useful changes or modifying what is being experienced.

SELF-LEARNING COMPUTER SYSTEMS

A self-learning computer can generate programs itself, enabling it to carry out new tasks. The methods the computer uses for this are of no importance now but we will return to the subject later. If we draw a comparison between the learning capacity of the computer and that of humans, we find ad unexpected discrepancy. The computer can do both more and less than a human; to the computer, comparing millions of pieces of data in a couple of minutes is simple. No human can do that. On the other hand it finds things that humans consider easy, such as recognizing a face or baking a cake, extremely difficult.

If we are going to use the learning capacity of computers for the solutions of problems;-we will have to restrict the problems to those areas in which computers specialize. At the moment, the computer's greatest power lies in its speed and accuracy; its greatest limitation is lack of creativity. A computer can solve simple puzzles, but not real problems.

Finding patterns in a marketing database with millions of records is relatively easy, but carrying out a task that really requires creativity and general knowledge of the world, such as solving a murder or drawing up a marketing plan, still presents the computer with insurmountable problems.

If we want to feed a certain problem to the computer, we first have to dissect it into manageable segments ourselves, and only learning problems that can be expressed in a limited number of strings of figures and symbols are suitable for this purpose yet, with the growing availability of large databases and the increasing need to interpret the information contained in these databases automatically, even the limited learning capabilities of current computers can prove of great value of an organization .

MACHINE LEARNING

Machine learning is the automation of a learning process and learning is tantamount to the construction of rules based on observations of environmental states and transitions. This is a broad field which includes not only learning from examples, but also reinforcement learning, learning with teacher, etc

A learning algorithm takes the data set and its accompanying information as input and returns a statement e.g. a concept representing the results of learning as output. Machine learning examines previous examples and their outcomes and learns how to reproduce these and make generalizations about new cases.

Generally a machine learning system does not use single observations of its environment but an entire finite set called the training set at once. This set contains examples i.e. observations coded in some machine readable form. The training set is finite hence not all concepts can be learned exactly.

Why machine learning is done?

- To understand and improve the efficiency of human learning.
- To discover new things or structure that is unknown to human beings.

• To fill in skeletal or computer specifications about a domain.

MACHINE LEARNING AND THE METHODOLOGY OF SCIENCE

The task of the modern scientist is to explain and to predict. Ideally the process of scientific research takes the form of a so-called empirical cycle

• Observation: we start with a number of observations.

• Analysis: we try to find patterns in these observations.

• Theory: if we have found some regularities, we formulate a theory (hypothesis) explaining the data.

Prediction: our theory will predict new phenomena that can be verified by new observations.

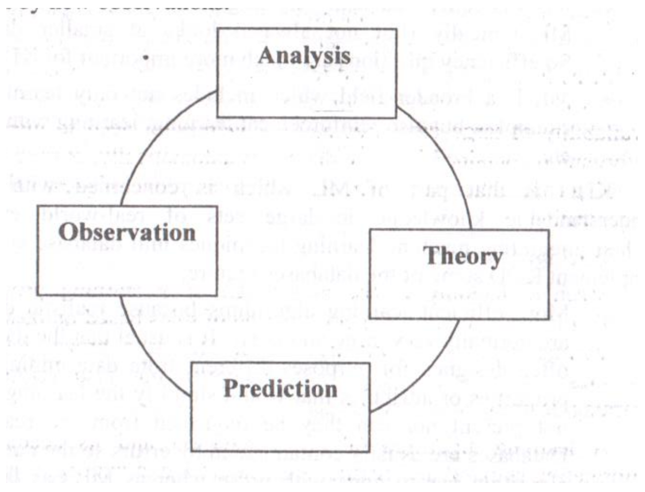


Figure — Empirical cycle of scientific research

In the last stage of the cycle there are two possibilities. Either our predictions are correct, in which case our theory is corroborated, or our predictions are wrong. In this latter case we have to analyze the new observations and try to come up with a new theory. So the whole process starts again, which is why we speak of an empirical cycle. The process goes on and on for ever, and we can refine out theories indefinitely. The same holds, apart form changes pf detail, for a manager who tries to analyze a market to develop new products or to optimize production. In any learning situation there is always some sort of empirical cycle.

In this century, the philosophy of science tends to defend the view that we can formulate hypotheses to explain empirical .observations but that we can never prove that they are true. Everything that science discovers has only temporary value.

DIFFERENCES BETWEEN DATA MINING AND MACHINE LEARNING

Knowledge Discovery in Databases (KDD) or Data Mining, and the part of Machine Learning (ML) dealing with learning Ii.om examples overlap in the algorithms used and the problems addressed.

The main differences are:

• KDD is concerned with finding understandable knowledge, while ML is concerned with improving performance of an agent. So training a neural network to balance a polo is part of ML, but not of KDD. However, there are efforts to extract knowledge from neural networks which are very relevant of KDD.

• KDD is concerned with very large, real-world databases, while ML typically (but not always) looks at smaller data sets. So efficiency questions are much more important for KDD.

• ML is a broader field which includes not only learning from examples, but also reinforcement learning, learning with teacher, etc.

KDD is that part of ML which is concerned with finding understandable knowledge in large sets of real-world examples. When integrating machine learning techniques into database systems to implement KDD some of the databases require:

• More efficient learning algorithms because realistic databases are normally very large and noisy. It is usual that the database is often designed for purposes different from data mining and so properties or attributes that would simplify the learning task are not present nor can they be requested from the real world. Databases are usually contaminated by errors so the data mining algorithm has to cope with noise whereas ML has laboratory type examples i.e. as near perfect as possible.

• More expressive representations for both data, e.g. tuples in relational databases, which represent instances of a problem domain, and knowledge, e.g. rules in a rule-based system, which can be used to solve users' problems in the domain, and the semantic information contained in the relational schemata.

CONCEPT LEARNING

There is a variety of different techniques to enable computers to learn concepts. A very important quality of good learning algorithms is that they learn consistent and complete definitions. A definition of a concept is complete is it recognizes all the instances of a concept; in our case, this means that it does not classify good examples of kangaroos as non-kangaroos.

A definition of a concept is consistent if it does not classify any negative examples as falling under the concept; in our case, this would mean that it would not recognize fish or birds as being kangaroos. As incomplete definition of the concept of the concept of a kangaroos would be too narrow and would fail to recognize all the kangaroos. An inconsistent definition of the concept of a kangaroo would be too broad, that is, it would classify some non-kangaroos as being kangaroos. Note that a definition can be both inconsistent and incomplete at the same time, although this would obviously be a very bad definition of a concept.

A very important element in machine learning is the language in which we express the hypothesis describing the concept. This language could be a specialized computer language like Prolog or Lisp, or a special form of knowledge representation using database tables. In the case of the kangaroo hypothesis, it might well be a set of attributes that could be stored in a database. The issues of the expressive power and the structure of the language in which we formulate our hypothesis are very important in machine learning.

DATA MINING AND THE DATA WAREHOUSE

Data Warehouse:

Definitions A data structure that is optimized for distribution. It collects and stores integrated sets of historical data from multiple operational systems and feeds them to one or more data marts. It may also provide end-user access to support enterprise views of data.

Why do we need Data Warehouse?

The data warehouse has specific characteristics that include the following:

Subject-Oriented: Information is presented according to specific subjects or areas of interest. not simply as computer files. Data is manipulated to provide information about a particular subject. For example, the SRDB is not simply made accessible to end-users, but is provided structure and organized according to the specific needs.

Integrated: A single source of information for and about understanding multiple areas of interest. The data warehouse provides one-stop shopping and contains information about a variety of subjects.

Non-Volatile: Stable information that doesn't change each time an operational process is executed. Information is consistent regardless of when the warehouse is accessed.

Time-Variant: Containing a history of the subject, as well as current information. Historical information is an important component of a data warehouse.

Designing decision support systems

The design of a decision support system differs considerably from that of an online transaction processing system. The main difference is that decision support systems are used only for queries. so their structure should be optimized for this use.

Hardware and software products of a decision support system

Within a data warehouse you need to satisfy specific hardware and software requirements in order to enable decision support to be successfully accomplished. The type of software you choose depends very much on the requirements of the end-user. Working in a client/server environment allows you great flexibility in choosing the appropriate software for end-users because each individual need can be catered for on a local workstation.

The only common element is the database, which must be completely optimized and provided with a number of functionalities to help speed up performance. Modern databases can also replicate parts of themselves to other databases. For data mining you can split the software into two parts: the first works with the algorithms on the database server and the second on the local workstation. The latter is mostly used to generate screens and reports for the end-user; in other words. It visualizes the outcome of the algorithms.

The hardware requirements depend on the type of data warehouse and techniques with which you want to work. A large data warehouse can contain hundreds of thousands of gigabytes and, in this case. it is important that the warehouse is designed by engineers with knowledge of both hardware and software. It is not always necessary to have a very' large database and a large database server for data mining- it depends on the volume of data involved and the techniques you wish to use. Not every database is able to provide all the techniques you need to set up a data mining environment, since it must have specific optimization tools, indexing facilities, and other techniques, like user-defined functions.

Integration with data mining

Integration of data mining in a decision support system is very helpful. The sole function of data warehouse is to supply the information needed to make adequate decisions. In some cases you can use standard structured query language (SQL) tools for decision support, but if you want to compare millions of records and do not know exactly the type of information you require, or if you want to find hidden data, then you have to turn to data mining.

There are several types of data mining techniques and each uses the computer in a specific way. For that reason it is important to understand the demands of the end-user so that you are able to build a proper data warehouse foe data mining. In many cases you will find that you need a separate computer for data mining; trying to mine operational data is almost impossible because there are different applications with different types of attributes and different data types but no historical data. With a data warehouse this problem does not exist-all the information has been transferred form the operational database to data ware house: furthermore, in many cases you can clean the data before commencing data mining.

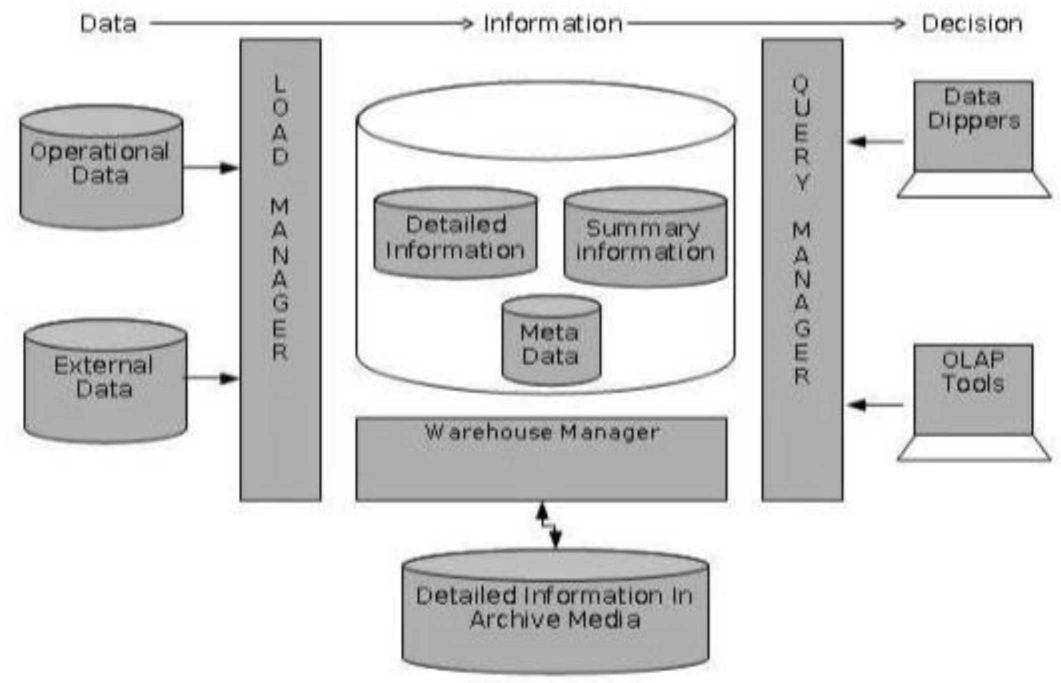


Figure — The relationship between operational data, data warehouse, and data marts

Client/server and data warehousing

Client/server server involves dispersing the software over several computers and creating an environment for the end-user so that it appears that each is working on just one system. The heavy load of graphical user interfaces or other visual techniques can be processed on this local machine and all the database tasks handled by a specific database server. In this way the database server can be completely optimized for the database. In some cases you can buy special databases that operate with a special type of hardware.

With client/server you only have to change the piece of software that is related to the end/user —the other application do not require alteration. Of all the techniques currently available on the market, client/server represents the best choice for building a data warehouse. Replication techniques are used to load the information from the operational database to the data warehouse.

The end-user can either visualize the work on the local workstation or connect up to a display server that has access to the data warehouse running on one or more database server. Data can be extracted to a local database system, and processed using that algorithms database. This is all possible with client/server techniques because, within a client/server environment, each computer is set up to fully optimize the end-user's application.

Two basic techniques are used to build a data warehouse, known as the 'top down' and the 'bottom up' approaches.

In the 'top down' approach, you first build a data warehouse for the complete organization, which will be a huge database where all the end-user information is stored, and from this select the information needed for your department or for local end-users.

In the 'bottom up' approach, smaller local data warehouses, known as data marts, are used by end-users at local level for their specific local requirements, such as for query purposes or for statistical analyses. These local end-users are not dependent on other data warehouses in the organization. A local data warehouse can be built in a very short period of time and can be managed at a departmental level, each data mart being completely optimized for particular tasks, such as data mining.

Multi-processing machines

A data mining environment has specific hardware requirements. Certain machine-learning tasks will involve the comparison of millions of records, thus placing a tremendous, burden on the system. If an end-user wants to compare large numbers of records within a very short period of time, the computer may need all its internal memory and all its processing power for this one task alone. In order to get a satisfactory answer at least two options are available: define the question by focusing on a limited number of records and attributes within a database, or move towards a multi-processing machines are needed for data mining projects; the end-user defines the records and attributes to be worked on and an extract from the original database is copied to a multi-processing machine. All the records are stored on its hard disk do that this machine can be used only for data mining. There are several types of multi-processing machines and we will describe the two most important ones :

- Symmetric multi-processing
- Massively parallel

With the symmetric multi-processing machine all the processors work on one computer, all are equal and they communicate via shared storage. Symmetric multi-processing machines share the same hard disk and the internal memory. Although processors share their internal coordination, this type of multi-processing is limited to a certain number of processors because the synchronization of the processors places a huge burden on the computer system; at the present time. Approximately twelve processors is the maximum. Because everything is shared, this type of symmetric multiprocessing can be used for data mining and, in many cases, this type of machine is sufficient for an organization's data mining activity.

The massively parallel machine is a computer where each processor has its own operating system. its own memory, and hard disk. Although each processor it's independent, communication between the systems is possible. In this type of environment one can work with thousands of processors; however, with each part of the software running on specific processor, it takes a great deal of time to assemble the software in the correct order .such a computer is very large data mining projects, since It is extremely powerful and has a very good response time.

Cost justification

It is difficult to give a cost justification for the implementation of a KDD environment. Implementing KDD in an organization involves issues like data warehousing and business process re-engineering, and cost justification for data mining is partly tied to the cost justification of these disciplines. However, we can say something about the effectiveness of data mining in a narrower sense.

Basically the cost of using machine-learning techniques to recognize patterns in data must be compared with the cost of human performing the same task.

Speed : A computer can easily read billions of records in a short Time

Complexity : There are certain tasks, in the field of calculation be made, and humans are not especially good at this.

Repetition : A computer does not become tired. One of the big advantages of implementing machine-learning algorithms on computers is that they can be repeated automatically as many times. This element of repetition is a particularly big issue in cost justification.

These three elements — speed, complexity, and repetition —indicate that it might take considerable effort to set up a mining environment.

Another way of looking at the cost justification for machine-learning techniques is found in the expert systems business. Expert systems are, quite simply, systems that contain the knowledge of specialists.

UNIT – II Knowledge Discovery Process

KNOWLEDGE DISCOVERY PROCESS

The knowledge discovery process consists of six stages:

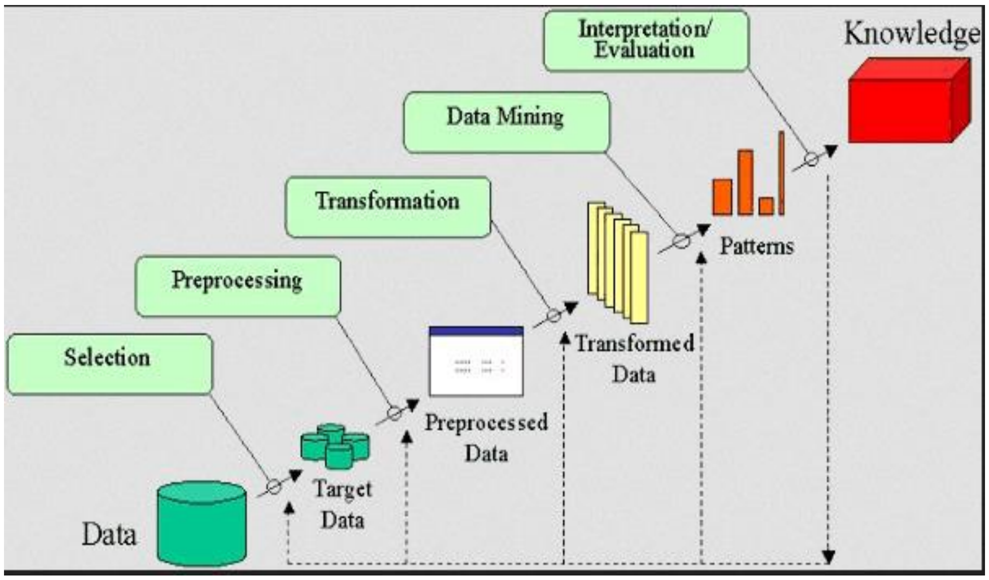
- Data selection
- Cleaning
- Enrichment
- Coding
- Data mining
- Reporting

Data selection

It is the process of retrieving relevant data from the database. In our example we start- with a rough data base containing records of subscription data for the magazines. It is a selection of operational data from the publisher's invoicing system and contains information about people who have subscribed to a magazine. The records consist of client number, name, address, date of subscription, and type of magazine. In order to facilitate the KDD process, a copy of this operational data is drawn and stored in a separate database.

Cleaning

A very important element in a cleaning operation is the de-duplication of records in the de-duplication of records .in a normal client database some clients will be represented by several records, although in many cases this will be the result of negligence, such as people making typing errors, or of clients moving from one place to another without notifying change of address. There are also cases in which people deliberately spell their names incorrectly or give incorrect information about themselves.



Coding

We lack vital data concerning Mr. King, so we choose to exclude this record from the final sample of course, this decision is questionable, because there may be a causal connection between the lack of information and certain purchasing behavior be Mr. King. For the moment we will suppose that we can omit this data without consequences for out final results. Next carry out a projection of the records. In this example we are nit interaed in the clients' name, since we just want to identify certain types of client. so their names are removed from the sample database.

DATA MINING

Data mining is not so much a single techniques as the idea that there is more knowledge hidden in the data than itself on the surface any technique that helps extract more out of your data is useful, so data mining techniques from quite a heterogeneous group. Although various different techniques are used for different purposes.

- Query tools
- Statistical techniques
- Visualization
- Online analytical processing (OLAP)
- Case-based learning (k-nearest neighbor)
- Decision trees
- Association rules
- Neural networks
- Genetic algorithms

PRELIMINARY ANALYSIS OF THE DATA SET USING TRADITIONAL QUERY TOOLS

The first step in a data mining project should always be a rough analysis of the data set using traditional query tools. Just by applying simple structured query language (SQL) to a data set, you can obtain a wealth of information. However. before we can apply more advanced pattern analysis algorithms, we need to know some basic aspects and structures of the data set. With SQL we can uncover only shallow data which is information that is easily accessible from the data set.

VISUALIZATION TECHNIQUES

Visualization techniques are a very useful method of discovering patterns in data sets, and may be used at the beginning of a data mining process to get a rough feeling of the quality of the data set and where patterns are to be found. Interesting possibilities are offered by object-oriented three dimensional tool kits, such as inventor, which enable the user to explore three-dimensional structures interactively. In the section below on decision trees we will give an example of the use of these tools for exploration of tree structures. Such techniques are developing rapidly: advanced graphical techniques in virtual reality enable people to wander through artificial data spaces, while historic development of data sets can be displayed as a king of animated movie. For most users, however, these advanced features are not accessible, and they have to rely on simple, graphical display techniques that are contained in the query tool or data mining tools they are using. These simple methods can provides us with a wealth of information. An elementary technique that can be of great value is the so-called scatter diagram; in this technique, information on two attributes is displayed in a Cartesian space. Scatter diagrams can be used to identify interesting sub-sets of the data sets so that we can focus on the rest of the data mining process. There is a whole field of research dedicated to the search for interesting projections of data sets —this is called projection pursuit.

LIKELIHOOD AND DISTANCE

There are other reasons to conceive records as points in a multi-dimensional data space. The space-metaphor is very useful in a data mining context. Using this metaphor we can determine the distance between two records in this data space: records that are close to each other are very alike, and records that are very far removed from each other represent individuals that have little in common. Our sample database contains attributes such age, income, and credit; these three attributes form a three-dimensional data space and we can analyze the distances between records in this space.

OLAP TOOLS

This idea of dimensionality can be expanded: a table with n independent attributes can be seen as an n-dimensional space. Managers generally ask questions that pre-suppose a multi-dimensional analysis-they don't want to know how much is sold(a zero-dimensional question) but they do want to know what type of magazines are sold in a designated area per month and to what age group (a four-dimensional question: product, area, purchase date, and age). nature is called multi-dimensional and such relationships cannot easily be analyzed when the table has the standard two-dimensional representation. We need to explore the relationship between several dimensions and standard relational databases are not very good at this. They identify records using keys and multi-dimensional relationships pre-suppose multiple keys, but there is a limit to the number of keys we can define effectively for a given table. There is, however, almost no end to the type of questions managers can formulate: one minute a manager might want sales data ordered by area, age, and income; the next minute, the same data ordered by credit and age and these preferably online using large data sets, OLAP tools were developed to solve these problems. These tools store their data in a special multi-dimensional format, often in memory, and a manager can ask any question at all, although the data cannot be updated. OLAP can be an important stage in a data mining process. There is however an important difference between data mining and OLAP tools: OLAP tools do not learn, they create no new knowledge, and they cannot search for new solutions. There is thus a fundamental difference between multi-dimensional knowledge one can extract from a database via data mining. Data mining is more powerful than OLAP. Another advantage is that data mining algorithms do not need a special form of storage since they can work directly on data stored in a relational database.

K-NEAREST NEIGHBOR

Records that are close to each other live in each other's neighborhood. The basic philosophy of k-nearest neighbor is 'do as your neighbors do.' K-nearest neighbor is not really a learning technique but more a search method. However, it is one of the purest search techniques, because the data set itself is used as our reference. Nevertheless, it is unable to create a theory of our domain that helps us understand its structure better, and its complexity can also prove a drawback. For instance. if we want to make a prediction for each element in the data set containing n records, then we have to compare each record with every other record. This leads to quadratic complexity, which is not desirable for large data sets. If we want to do a simple k-nearest neighbor analysis of a database with a million records, then we have to make a thousand billion comparisons. Such an approach does not scale up well, although there exist a lot of research results that help to speed up these search processes. In general, data mining algorithms should not have a complexity that is higher than n(log n) (where n is the number of records). In practice. we will use the simple k-nearest neighbor technique mainly on sub-samples or data sets of a limited size.

A problem in the application of the k-nearest neighbor process is the existence of tables containing fairly high number of attributes. If these attributes are independent of each other, this means that we are working with a high-dimensional search space, which is undesirable for several reasons. One of the main drawbacks of high-dimensional spaces is that our normal intuitions abut what space is failed to apply. A three-dimensional space with an equal distribution of a million data points is fairly crowed and we obtain good results from nearest neighbor queries.

DECISION TREES

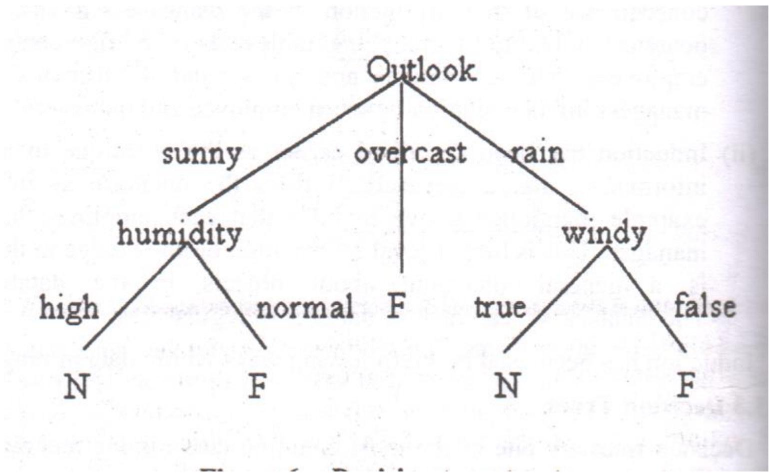
Decision trees are one of the most common data mining techniques and are by far the most popular in tools aimed at the business user. They are easy to set up, their results are understandable by an end-user, they can address a wide range of classification problems, they are robust in the face of different data distributions and formats, and they are effective in analyzing large numbers of fields.

A decision tree algorithm works by splitting a data set in order to build a model that successfully classifies each record in terms of a target field or variable. An example is a decision tree which classifies a data set according to whether customers did or did not buy a particular product.

The most common types of decision tree algorithm are CHAID, CART and C4.5. CHAID (Chi-square automatic interaction detection) and CART (Classification and Regression Trees) were developed by statisticians. CHAID can produce tree with multiple sub-nodes for each split. CART requires less data preparation than CHAID, but produces only two-way splits. C4.5 comes from the world of machine learning, and is based on information theory.

In order to generate a decision tree from the training set of data we need to split the data into progressively smaller subsets. Each iteration considers the data in only one node. The first iteration considers the root node that contains all the data. Subsequent iterations work on derivative nodes that will contain subsets of the data.

At each iteration, we need to choose the independent variable that most effectively splits the data. This means that the subsets produced by splitting the data according to the value of the independent variable should be as -homogeneous- as possible, with respect to the dependent variable.



ASSOCIATION RULES

It is a model that identifies specific types of data associations. These associations are often used in the retail sales community to identify items that are frequently purchased together. Data can he mined to identify associations.

It is a model that identifies specific types of data associations. These associations are often used in the retail sales community to identify items that are frequently purchased together. Data can be mined to identify associations.

Definition: Given a set of items $I =\{i_1,i_2,---i_m\}$ and a database of transaction $D = =\{t_1,t_2,---t_n\}$ where $t_i =\{i_{i1},i_{i2},---i_{ik}\}$ and i_{ij} E I, an association rule is an implication of the form $X \rightarrow Y$ where X,Y c I are sets of items called item sets and X n Y = 0.

Large Item sets

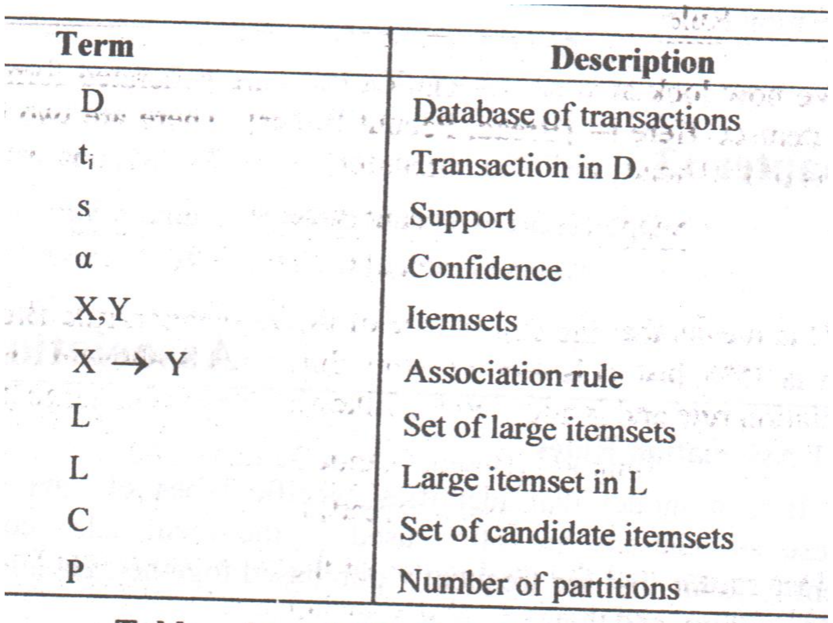
The most common approach to finding association rules is to break up the problem into two parts:

1. Find large itemsets as defined in Definition

2. Generate rules from frequent itemsets.

An itemsets is any subset of the set of all items, i.

Definition: A large (frequent) itemset is an itemset whose number of occurrences is above a threshold, s. We use the notation L to indicate the complete set of large itemsets and 1 to indicate a specific large itemset. Once the large itemsets have been found, we know- that any interesting association rule, X Y, must have x U y in this set of frequent itemsets. Note that the subset of any large itemset is also large. Because of the large number of notations used in association rule, algorithms, we summarize them in table. When a specific term has a subscript, this indicates the size of the set being considered. For example, 1k is a large itemset of size k. Some algorithms divided the set of transactions into partitions. In this case, we use P to indicate the number of partitions and a superscript to indicate which partition. For example, D' is the i^{th} partition of D.



Term	Description
D	Database of transactions
t_i	Transaction in D
s	Support
α	Confidence
X,Y	Itemsets
$X \rightarrow Y$	Association rule
L	Set of large itemsets
L	Large itemset in L
C	Set of candidate itemsets
P	Number of partitions

Table – Association Rule Notation

NEURAL NETWORKS

Neural networks are an approach to computing that involves developing mathematical structures with the ability to learn. The methods are the result of academic investigations to model nervous system learning. Neural networks have the remarkable ability to derive meaning from complicated or imprecise data and can be used to extract patterns and detect trends that arc too complex to be noticed by either humans or other computer techniques. There are several difference forms of neural network.

- Perceptrons
- Back propagation networks
- Kohonen self-organizing map

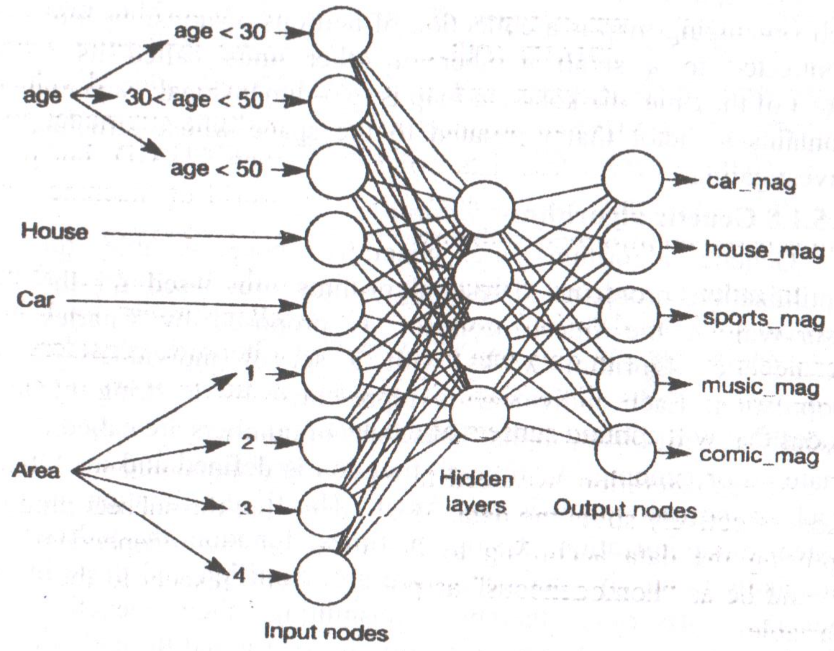


Figure — Neural network architecture for simple data mining

A perceptron consists of simple three-layered network with input units called photo-receptors; intermediate units called associators, and output units called responders. The perceptron could learn simple categories and thus could be used to perform simple classification tasks.

A major improvement was the introduction of hidden layers in the so called hack propagation networks.

A back propagation network not only has input and output nodes, but also a set of intermediate layers with hidden nodes. In its initial stage a back propagation network, expose it to a training set input data. The input nodes are wholly interconnected to the hidden nodes, and the nodes are wholly interconnected to the output nodes. In an untrained network the branches between the nodes have equal weights. During the training stage the network receives examples of input and output pairs corresponding to records in the database, and adapts the weights of the different branches until all the inputs match the appropriate outputs.

In 1981 Tuevo Kohonen demonstrated a completely different version of neural networks that is currently known as Kohonen's self-organizing maps. These neural networks can be seen as the artificial counterparts of maps that exist in several places in the brain, such as visual maps, maps of the spatial possibilities of limbs, and so on. A Kohonen self-organizing map is a collection of neurons of units, each of which is connected to a small number of other units called its neighbors. Most of the time. the kohonen map is two-dimensional; each node or unit contains a factor that is related to the space whose structure we are investigating.

<u>GENETIC ALGORITHMS</u>

Genetic algorithms are generally more suitable for solving optimization problems. These algorithms are based on the nature's -survival of the fittest" concept as proposed by Charles Darwin. A genetic algorithm works with a set of individuals forming a population. Each individual is represented by a string of numbers. usually a set of binary digits. These sets of numbers are called the genetic material or genome. A "fitness" function is defined and the -fitness" of each of the individuals is calculated. The fitness function indicates the extent of optimization. Using the fitness function, the individuals who are likely to survive and reproduce are identified and these individuals generate offspring, thereby transmitting their genetic material. The genetic material of the offspring is determined through a process of selection and crossover. The fitness of each of the new individuals in the population is examined and the cycle is repeated. In addition to the selection and crossover, another operation called -mutation" is also applied to selected individuals of the population. Generally, the mutation operation is carried out by interchanging a randomly selected bit of the genome string. Most often, the result of a mutation is harmful and the frequency is kept very low in the algorithm. Nevertheless, the mutation could result in a very significant increase in the fitness of the individual. There is a randomness associated in each of the operations namely, selection, crossover and mutation. In general, the overall fitness of the population increases after a few cycles. Still, the genetic algorithms do not always produce a global optimal solution. but they often produce a "near" optimal solution very quickly.

'Mere are many instances where data mining can be applied very effectively. It can be used very effectively to identify the customers who are most likely to purchase a particular product. It can be used to identify who can be targeted for cross selling or up selling. Market segmentation is an effective application of data mining. It will help in minimizing churn. It has been used in quality control and for minimizing wastage in production.

Reporting

Reporting the results of data mining can take many forms. The illustrations already described in this chapter have given a good idea of the options available. In general, one can use any report writer or graphical tool to make the results of the process accessible. The material in this chapter had also provided a good indication of the interactive character of the KDD process: there is constant interplay between the selection of data, cleaning, data mining, and the reporting of results.

KDD ENVIRONMENT

Different forms of knowledge

The key issue in KDD is to realize that there is more information hidden in your data than you are able to distinguish at first sight. In fact. in data mining we distinguish four different types of knowledge that can be extracted from the data :

1. Shallow knowledge: This is information that can be easily retrieved from databases using a query tool such as structured query language (SQL).

2. Multi-dimensional knowledge: this is information that can be analyzed using online analytical processing tools. With OLAP tools you have the ability to rapidly explore all sorts of clustering's and different orderings of the data but it is important to realize that most of the things you can do with an OLAP tool can also be done using SQL. The advantage of OLAP tools is that they are optimized for this kind of search and analysis operation. However, OLAP is not as powerful as data mining: it cannot search for optimal solutions.

3. Hidden knowledge: this is data can be found relatively easily be using pattern recognition o machine-learning algorithms. Again one could use SQL to find these patterns but this would probably prove extremely time-consuming. A pattern recognition algorithm could find regularities in a database in minutes or at most a couple of hours. whereas you would have to spend months using SQL to achieve the same result.

4. Deep knowledge: this is information that is stored I the database but can only be located if we have a clue that tells us where to look.

KDD Environment

The goal of a KDD process is to obtain an ever-increasing and better understanding of the changing environment of the organization. A KDD environment supports the data mining process, but this process is of involved that is neither realistic nor desirable to try to support it with just one generic tool. Rather, one needs a suite of tools that is carefully selected and tuned specifically for each organization utilizing data mining. Some tools being marketed at present as data mining tools deserves this name only in the sense that one can call a hammer a ship-building tool.

Ten golden rules

It is customary in the computer industry to formulate rules of thumb that help information technology (IT) specialists to apply new developments. In this section we give ten golden rules for setting up a reliable data mining environment. These can be used by IT professionals to select tools and to judge the proposals of vendors.

1. Support extremely large data sets

Data mining deals with extremely large data sets consisting in some cases of billions of records, and without proper platforms to store and handle these volumes of data, no reliable data mining is possible parallel servers with databases optimized for DSS-oriented queries are useful. Fast and flexible access to large data sets is of vital importance so too is the possibility of storing intermediate results.

2. Support hybrid learning

Learning tasks can be divided into three areas:

- classification tasks
- knowledge engineering tasks
- problem-solving tasks

Not all algorithms do equally well in all these areas: in complex data mining projects, such as fraud detection or customer profiling, various pattern recognition and learning algorithms (neural networks, genetic algorithms, statistical techniques, association rules, rule discovery, and so on) are needed .

3. Establish a data warehouse

A data warehouse contains historic data and is subject oriented and static, that is, users do not update the data but it is created on a regular time-frame on the basis of the operational data of an organization. It is thus obvious that a data warehouse is an ideal starting point for a data milling process, since data mining depends heavily on the permanent availability of historic data, and in this sense a data warehouse could be regarded as indispensable,

4. Introduce data cleaning facilities

Even when a data warehouse is in operation, the data is certain to contain all sorts of pollution and as the data mining process continues more subtle forms of pollution will be identified. Special tools for cleaning data are necessary, and some advanced tools are available, especially in the field of de-duplication of client files. Other cleaning techniques are only just emerging from research laboratories.

5. Facilitate working with dynamic coding

Creative coding is the heart of the KDD process. The environment should enable the user to experiment with different coding schemes, store partial results, make attributes discrete, create time series out of historic data, select random sub-samples separate test sets, and so on. A project management that keeps track of the genealogy of different samples and tables as well as of the semantics and transformations of the different attributes is vital.

6. Integrate with DSS

Data mining looks for hidden data that cannot easily be found using normal query techniques. Nevertheless, a KDD process always starts with traditional DSS activities, and from there you zoom in on interesting parts of the data set.

7. Choose extendible architecture

New techniques for pattern recognition and machine learning are under development and we also see numerous new developments in the database area. It is advisable to choose an architecture that enables you to integrate new tools at later stages. Object-oriented technology, such as CORBA, typically facilitates this kind of flexibility.

8. Support heterogeneous databases

Not all the necessary data is necessarily to be found in the data warehouse. Sometimes you will need to enrich the data warehouse with information from unexpected sources, such as information) brokers, or with operational data that is not stored in your regular data warehouse. In order to facilitate this, the data mining environment must support a variety of interfaces: hierarchical databases, flat files, various relational databases, and object-oriented database systems.

9. Introduce client/server architecture

A data mining environment needs extensive reporting facilities. Some developments, such as data landscapes, point in direction of highly interactive graphic environments but database servers are not very suitable for this task. Discovery jobs need to be processed by large data mining servers, while further refinement and reporting will take place on a client. Separating the data mining activities on the servers from the clients is vital for good performance. Client/server is a much more flexible system which moves the burden of visualization and graphical techniques from your server to the local machine.

10. Introduce cache optimization

Leaning and pattern recognition algorithms that operate on databases often need very special and frequent access to the data. Usually it is either impossible or impractical to store the data in separate tables or to cache large portions in internal memory. The learning algorithms in a data mining environment should be optimized for this type of data access.

UNIT – III Data Ware house – Architecture

DATA WAREHOUSE

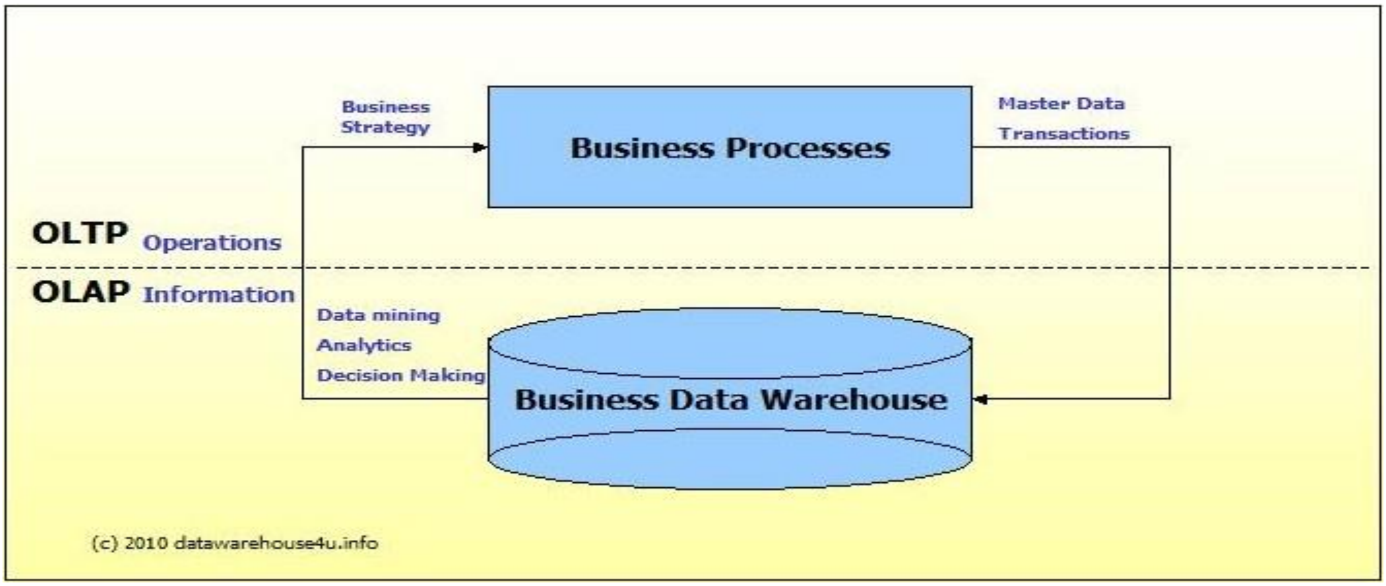
A data warehouse is a:

- subject-oriented
- integrated
- timevarying
- non-volatile collection of data in support of the management's decision-making process.

A data warehouse is a centralized repository that stores data from multiple information sources and transforms them into a common, multidimensional data model for efficient querying and analysis.

OLTP vs. OLAP

We can divide IT systems into transactional (OLTP) and analytical (OLAP). In general we can assume that OLTP systems provide source data to data warehouses, whereas OLAP systems help to analyze it.



OLTP (On-line Transaction Processing) is characterized by a large number of short on-line transactions (INSERT, UPDATE, DELETE). The main emphasis for OLTP systems is put on very fast query processing, maintaining data integrity in multi-access environments and an effectiveness measured by number of transactions per second. In OLTP database there is detailed and current data, and schema used to store transactional databases is the entity model (usually 3NF).

OLAP (On-line Analytical Processing) is characterized by relatively low volume of transactions. Queries are often very complex and involve aggregations. For OLAP systems a response time is an effectiveness measure. OLAP applications are widely used by Data Mining techniques. In OLAP database there is aggregated, historical data, stored in multi-dimensional schemas (usually star schema).

The following table summarizes the major differences between OLTP and OLAP system design.

	OLTP System Online Transaction Processing (Operational System)	**OLAP System Online Analytical Processing (Data Warehouse)**
Source of data	Operational data; OLTPs are the original source of the data.	Consolidation data; OLAP data comes from the various OLTP Databases
Purpose of data	To control and run fundamental business tasks	To help with planning, problem solving, and decision support
What the data	Reveals a snapshot of ongoing business processes	Multi-dimensional views of various kinds of business activities
Inserts and Updates	Short and fast inserts and updates initiated by end users	Periodic long-running batch jobs refresh the data
Queries	Relatively standardized and simple queries Returning relatively few records	Often complex queries involving aggregations
Processing Speed	Typically very fast	Depends on the amount of data involved; batch data refreshes and complex queries may take many hours; query speed can be improved by creating indexes
Space Requirements	Can be relatively small if historical data is archived	Larger due to the existence of aggregation structures and history data; requires more indexes than OLTP
Database Design	Highly normalized with many tables	Typically de-normalized with fewer tables; use of star and/or snowflake schemas
Backup and Recovery	Backup religiously; operational data is critical to run the business, data loss is likely to entail significant monetary loss and legal liability	Instead of regular backups, some environments may consider simply reloading the OLTP data as a recovery method

DATA WAREHOUSE ARCHITECTURE

The business analyst get the information from the data warehouses to measure the performance and make critical adjustments in order to win over other business holders in the market. Having a data warehouse offers the following advantages −

- Since a data warehouse can gather information quickly and efficiently, it can enhance business productivity.
- A data warehouse provides us a consistent view of customers and items, hence, it helps us manage customer relationship.
- A data warehouse also helps in bringing down the costs by tracking trends, patterns over a long period in a consistent and reliable manner.

To design an effective and efficient data warehouse, we need to understand and analyze the business needs and construct a **business analysis framework**. Each person has different views regarding the design of a data warehouse. These views are as follows −

- **The top-down view** − This view allows the selection of relevant information needed for a data warehouse.
- **The data source view** − This view presents the information being captured, stored, and managed by the operational system.
- **The data warehouse view** − This view includes the fact tables and dimension tables. It represents the information stored inside the data warehouse.
- **The business query view** − It is the view of the data from the viewpoint of the end-user.

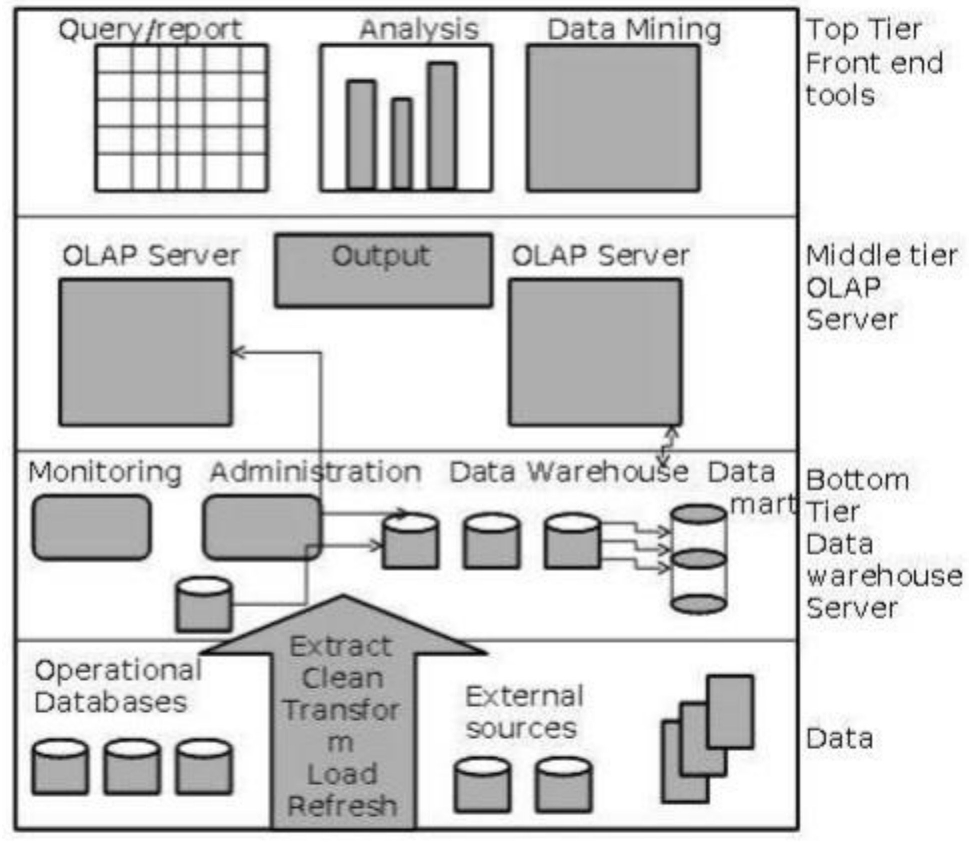
Three-Tier Data Warehouse Architecture

Generally a data warehouses adopts a three-tier architecture. Following are the three tiers of the data warehouse architecture.

- **Bottom Tier** − The bottom tier of the architecture is the data warehouse database server. It is the relational database system. We use the back end tools and utilities to feed data into the bottom tier. These back end tools and utilities perform the Extract, Clean, Load, and refresh functions.
- **Middle Tier** − In the middle tier, we have the OLAP Server that can be implemented in either of the following ways.

  - By Relational OLAP (ROLAP), which is an extended relational database management system. The ROLAP maps the operations on multidimensional data to standard relational operations.
  - By Multidimensional OLAP (MOLAP) model, which directly implements the multidimensional data and operations.
- **Top-Tier** − This tier is the front-end client layer. This layer holds the query tools and reporting tools, analysis tools and data mining tools.

The following diagram depicts the three-tier architecture of data warehouse −



Data Warehouse Models

From the perspective of data warehouse architecture, we have the following data warehouse models −

- Virtual Warehouse
- Data mart
- Enterprise Warehouse

Virtual Warehouse

The view over an operational data warehouse is known as a virtual warehouse. It is easy to build a virtual warehouse. Building a virtual warehouse requires excess capacity on operational database servers.

Data Mart

Data mart contains a subset of organization-wide data. This subset of data is valuable to specific groups of an organization.

In other words, we can claim that data marts contain data specific to a particular group. For example, the marketing data mart may contain data related to items, customers, and sales. Data marts are confined to subjects.

Points to remember about data marts −

- Window-based or Unix/Linux-based servers are used to implement data marts. They are implemented on low-cost servers.
- The implementation data mart cycles is measured in short periods of time, i.e., in weeks rather than months or years.
- The life cycle of a data mart may be complex in long run, if its planning and design are not organization-wide.
- Data marts are small in size.
- Data marts are customized by department.
- The source of a data mart is departmentally structured data warehouse.
- Data mart are flexible.

Enterprise Warehouse

- An enterprise warehouse collects all the information and the subjects spanning an entire organization
- It provides us enterprise-wide data integration.
- The data is integrated from operational systems and external information providers.
- This information can vary from a few gigabytes to hundreds of gigabytes, terabytes or beyond.

Load Manager

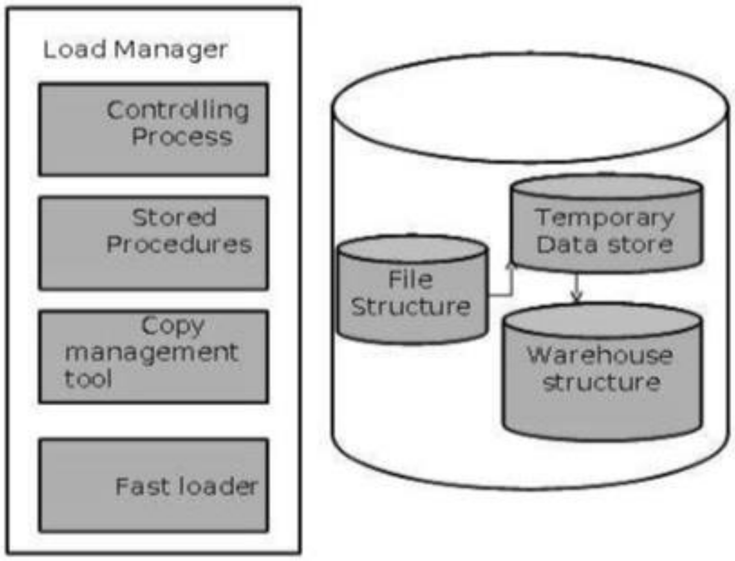
This component performs the operations required to extract and load process.

The size and complexity of the load manager varies between specific solutions from one data warehouse to other.

Load Manager Architecture

The load manager performs the following functions −

- Extract the data from source system.
- Fast Load the extracted data into temporary data store.
- Perform simple transformations into structure similar to the one in the data warehouse.



Extract Data from Source

The data is extracted from the operational databases or the external information providers. Gateways is the application programs that are used to extract data. It is supported by underlying DBMS and allows client program to generate SQL to be executed at a server. Open Database Connection(ODBC), Java Database Connection (JDBC), are examples of gateway.

Fast Load

- In order to minimize the total load window the data need to be loaded into the warehouse in the fastest possible time.
- The transformations affects the speed of data processing.
- It is more effective to load the data into relational database prior to applying transformations and checks.
- Gateway technology proves to be not suitable, since they tend not be performant when large data volumes are involved.

Simple Transformations

While loading it may be required to perform simple transformations. After this has been completed we are in position to do the complex checks. Suppose we are loading the EPOS sales transaction we need to perform the following checks:

- Strip out all the columns that are not required within the warehouse.
- Convert all the values to required data types.

Warehouse Manager

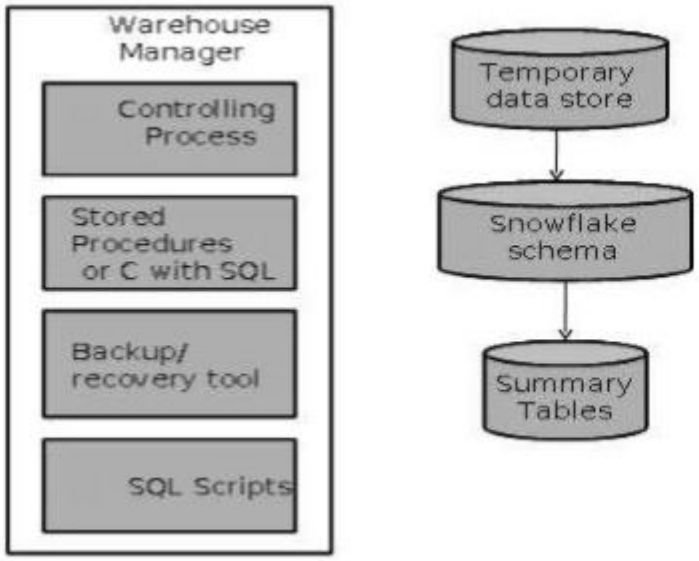
A warehouse manager is responsible for the warehouse management process. It consists of third-party system software, C programs, and shell scripts.

The size and complexity of warehouse managers varies between specific solutions.

Warehouse Manager Architecture

A warehouse manager includes the following −

- The controlling procfess
- Stored procedures or C with SQL
- Backup/Recovery tool
- SQL Scripts



Operations Performed by Warehouse Manager

- A warehouse manager analyzes the data to perform consistency and referential integrity checks.
- Creates indexes, business views, partition views against the base data.

- Generates new aggregations and updates existing aggregations. Generates normalizations.
- Transforms and merges the source data into the published data warehouse.
- Backup the data in the data warehouse.
- Archives the data that has reached the end of its captured life.

Note – A warehouse Manager also analyzes query profiles to determine index and aggregations are appropriate.

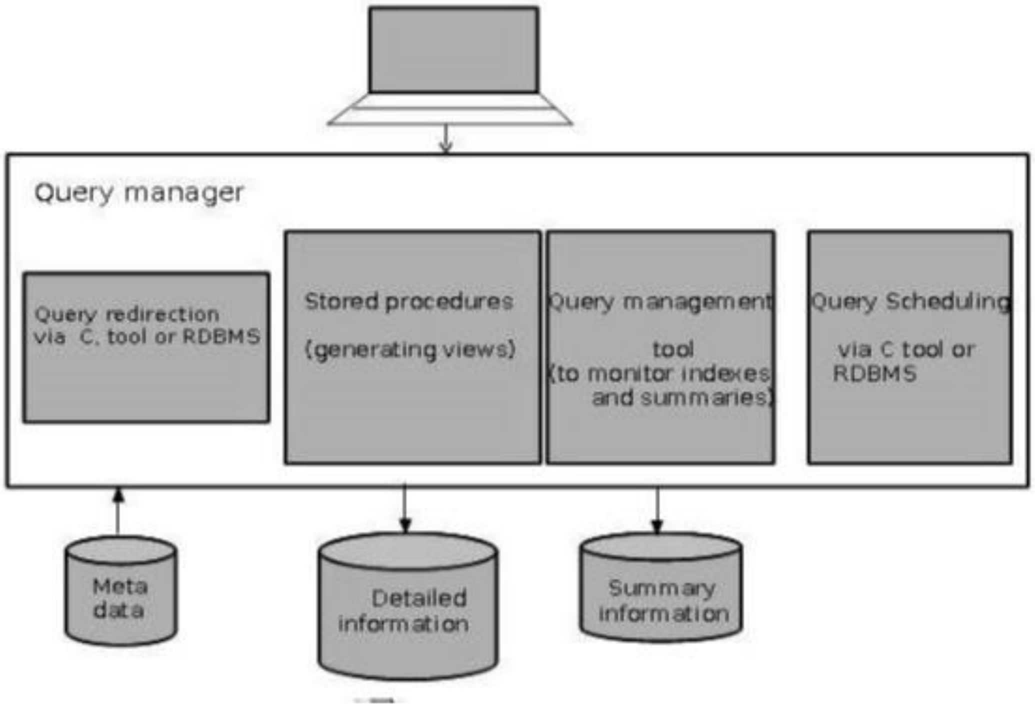
Query Manager

- Query manager is responsible for directing the queries to the suitable tables.
- By directing the queries to appropriate tables, the speed of querying and response generation can be increased.
- Query manager is responsible for scheduling the execution of the queries posed by the user.

Query Manager Architecture

The following screenshot shows the architecture of a query manager. It includes the following:

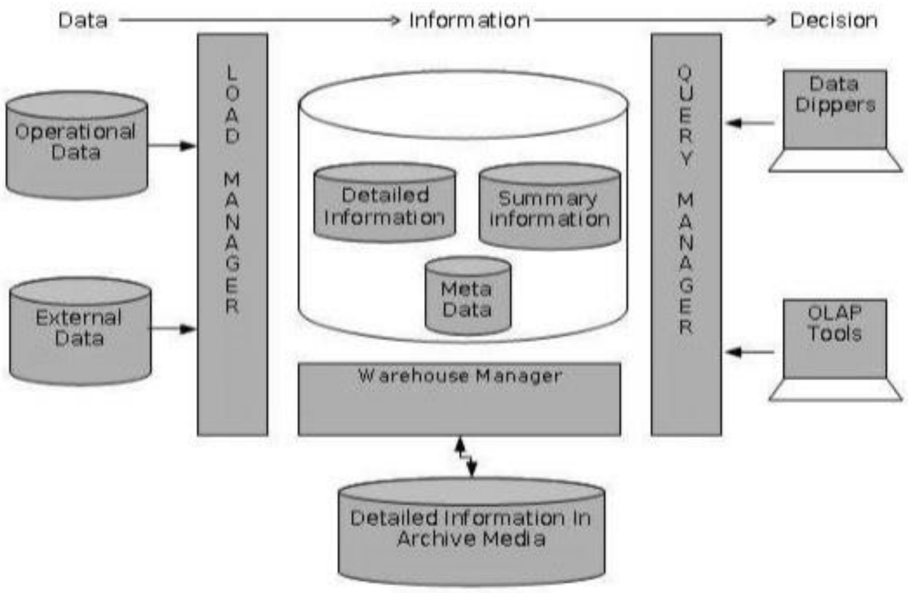
- Query redirection via C tool or RDBMS
- Stored procedures
- Query management tool
- Query scheduling via C tool or RDBMS
- Query scheduling via third-party software


Query manager
Query redirection via C, tool or RDBMS
Stored procedures (generating views)
Query management tool (to monitor indexes and summaries)
Query Scheduling via C tool or RDBMS
Meta data
Detailed information
Summary information

Detailed Information

Detailed information is not kept online, rather it is aggregated to the next level of detail and then archived to tape. The detailed information part of data warehouse keeps the detailed information in the starflake schema. Detailed information is loaded into the data warehouse to supplement the aggregated data.

The following diagram shows a pictorial impression of where detailed information is stored and how it is used.



Note − If detailed information is held offline to minimize disk storage, we should make sure that the data has been extracted, cleaned up, and transformed into starflake schema before it is archived.

Summary Information

Summary Information is a part of data warehouse that stores predefined aggregations. These aggregations are generated by the warehouse manager. Summary Information must be treated as transient. It changes on-the-go in order to respond to the changing query profiles.

The points to note about summary information are as follows −

- Summary information speeds up the performance of common queries.
- It increases the operational cost.
- It needs to be updated whenever new data is loaded into the data warehouse.
- It may not have been backed up, since it can be generated fresh from the detailed information.

SYSTEM PROCESS

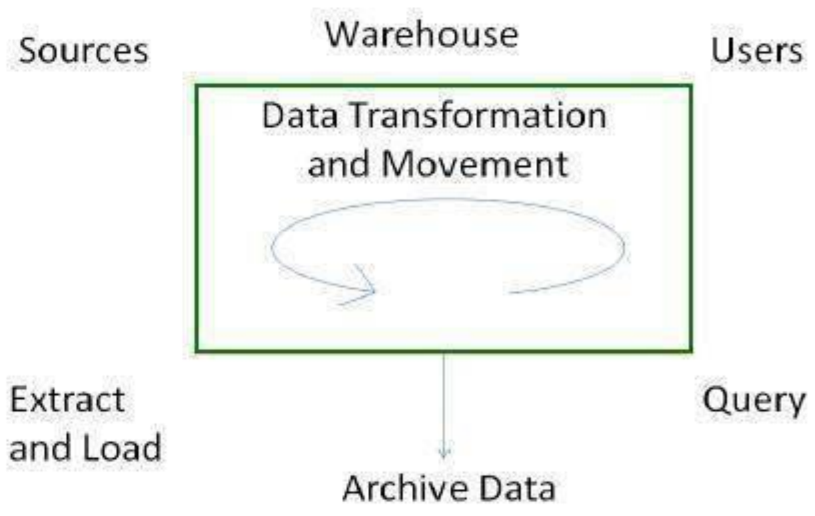
We have a fixed number of operations to be applied on the operational databases and we have well-defined techniques such as **use normalized data**, **keep table small**, etc. These techniques are suitable for delivering a solution. But in case of decision-support systems, we do not know what query and operation needs to be executed in future. Therefore techniques applied on operational databases are not suitable for data warehouses.

In this chapter, we will discuss how to build data warehousing solutions on top open-system technologies like Unix and relational databases.

Process Flow in Data Warehouse

There are four major processes that contribute to a data warehouse −

- Extract and load the data.
- Cleaning and transforming the data.
- Backup and archive the data.
- Managing queries and directing them to the appropriate data sources.



Extract and Load Process

Data extraction takes data from the source systems. Data load takes the extracted data and loads it into the data warehouse.

Note – Before loading the data into the data warehouse, the information extracted from the external sources must be reconstructed.

Controlling the Process

Controlling the process involves determining when to start data extraction and the consistency check on data. Controlling process ensures that the tools, the logic modules, and the programs are executed in correct sequence and at correct time.

When to Initiate Extract

Data needs to be in a consistent state when it is extracted, i.e., the data warehouse should represent a single, consistent version of the information to the user.

For example, in a customer profiling data warehouse in telecommunication sector, it is illogical to merge the list of customers at 8 pm on Wednesday from a customer database with the customer subscription events up to 8 pm on Tuesday. This would mean that we are finding the customers for whom there are no associated subscriptions.

Loading the Data

After extracting the data, it is loaded into a temporary data store where it is cleaned up and made consistent.

Note − Consistency checks are executed only when all the data sources have been loaded into the temporary data store.

Clean and Transform Process

Once the data is extracted and loaded into the temporary data store, it is time to perform Cleaning and Transforming. Here is the list of steps involved in Cleaning and Transforming −

- Clean and transform the loaded data into a structure
- Partition the data
- Aggregation

Clean and Transform the Loaded Data into a Structure

Cleaning and transforming the loaded data helps speed up the queries. It can be done by making the data consistent −

- within itself.
- with other data within the same data source.
- with the data in other source systems.
- with the existing data present in the warehouse.

Transforming involves converting the source data into a structure. Structuring the data increases the query performance and decreases the operational cost. The data contained in a data warehouse must be transformed to support performance requirements and control the ongoing operational costs.

Partition the Data

It will optimize the hardware performance and simplify the management of data warehouse. Here we partition each fact table into multiple separate partitions.

Aggregation

Aggregation is required to speed up common queries. Aggregation relies on the fact that most common queries will analyze a subset or an aggregation of the detailed data.

Backup and Archive the Data

In order to recover the data in the event of data loss, software failure, or hardware failure, it is necessary to keep regular back ups. Archiving involves removing the old data from the system in a format that allow it to be quickly restored whenever required.

For example, in a retail sales analysis data warehouse, it may be required to keep data for 3 years with the latest 6 months data being kept online. In such as scenario, there is often a requirement to be able to do month-on-month comparisons for this year and last year. In this case, we require some data to be restored from the archive.

Query Management Process

This process performs the following functions −

- manages the queries.
- helps speed up the execution time of queris.
- directs the queries to their most effective data sources.
- ensures that all the system sources are used in the most effective way.
- monitors actual query profiles.

The information generated in this process is used by the warehouse management process to determine which aggregations to generate. This process does not generally operate during the regular load of information into data warehouse.

PROCESS ARCHITECTURE

Process managers are responsible for maintaining the flow of data both into and out of the data warehouse. There are three different types of process managers −

- Load manager
- Warehouse manager
- Query manager

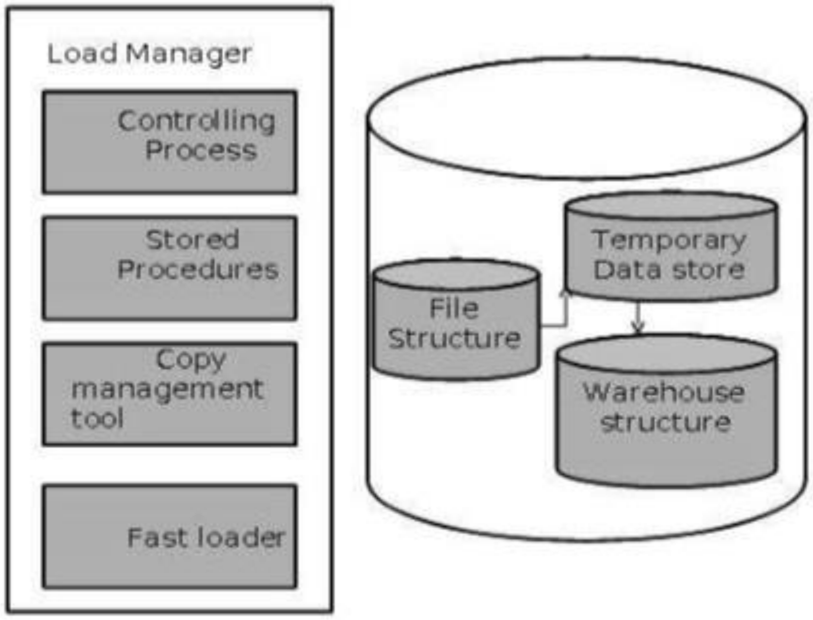
Data Warehouse Load Manager

Load manager performs the operations required to extract and load the data into the database. The size and complexity of a load manager varies between specific solutions from one data warehouse to another.

Load Manager Architecture

The load manager does performs the following functions −

- Extract data from the source system.
- Fast load the extracted data into temporary data store.
- Perform simple transformations into structure similar to the one in the data warehouse.



Extract Data from Source

The data is extracted from the operational databases or the external information providers. Gateways are the application programs that are used to extract data. It is supported by underlying DBMS and allows the client program to generate SQL to be executed at a server. Open Database Connection (ODBC) and Java Database Connection (JDBC) are examples of gateway.

Fast Load

- In order to minimize the total load window, the data needs to be loaded into the warehouse in the fastest possible time.
- Transformations affect the speed of data processing.
- It is more effective to load the data into a relational database prior to applying transformations and checks.
- Gateway technology is not suitable, since they are inefficient when large data volumes are involved.

Simple Transformations

While loading, it may be required to perform simple transformations. After completing simple transformations, we can do complex checks. Suppose we are loading the EPOS sales transaction, we need to perform the following checks −

- Strip out all the columns that are not required within the warehouse.
- Convert all the values to required data types.

Warehouse Manager

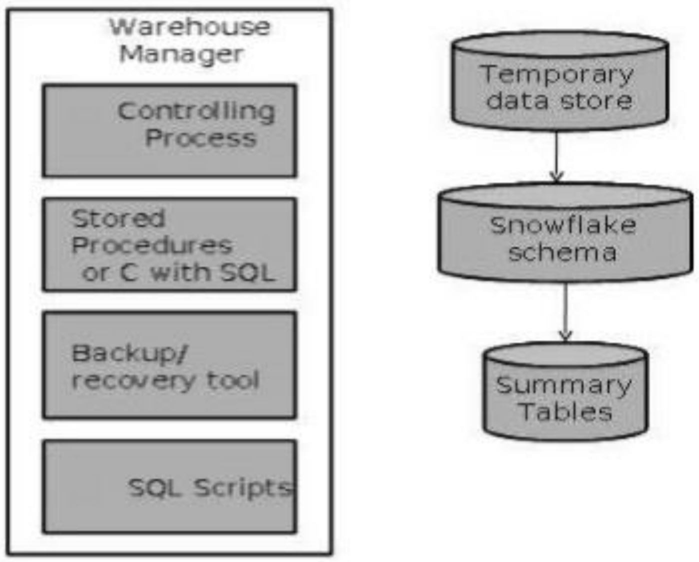
The warehouse manager is responsible for the warehouse management process. It consists of a third-party system software, C programs, and shell scripts. The size and complexity of a warehouse manager varies between specific solutions.

Warehouse Manager Architecture

A warehouse manager includes the following −

- The controlling process
- Stored procedures or C with SQL
- Backup/Recovery tool

- SQL scripts



Functions of Warehouse Manager

A warehouse manager performs the following functions −

- Analyzes the data to perform consistency and referential integrity checks.
- Creates indexes, business views, partition views against the base data.
- Generates new aggregations and updates the existing aggregations.
- Generates normalizations.
- Transforms and merges the source data of the temporary store into the published data warehouse.
- Backs up the data in the data warehouse.
- Archives the data that has reached the end of its captured life.

Note − A warehouse Manager analyzes query profiles to determine whether the index and aggregations are appropriate.

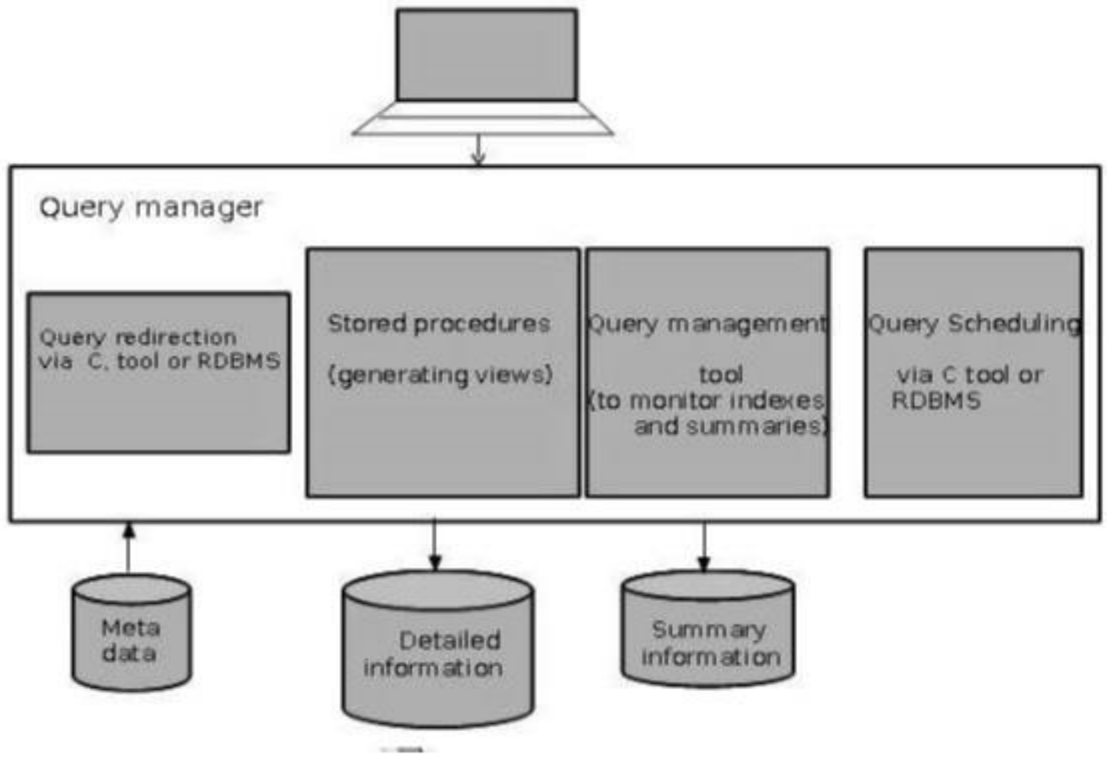
Query Manager

The query manager is responsible for directing the queries to suitable tables. By directing the queries to appropriate tables, it speeds up the query request and response process. In addition, the query manager is responsible for scheduling the execution of the queries posted by the user.

Query Manager Architecture

A query manager includes the following components −

- Query redirection via C tool or RDBMS
- Stored procedures
- Query management tool
- Query scheduling via C tool or RDBMS
- Query scheduling via third-party software


Query manager
Query redirection via C, tool or RDBMS
Stored procedures (generating views)
Query management tool (to monitor indexes and summaries)
Query Scheduling via C tool or RDBMS
Meta data
Detailed information
Summary information

Functions of Query Manager

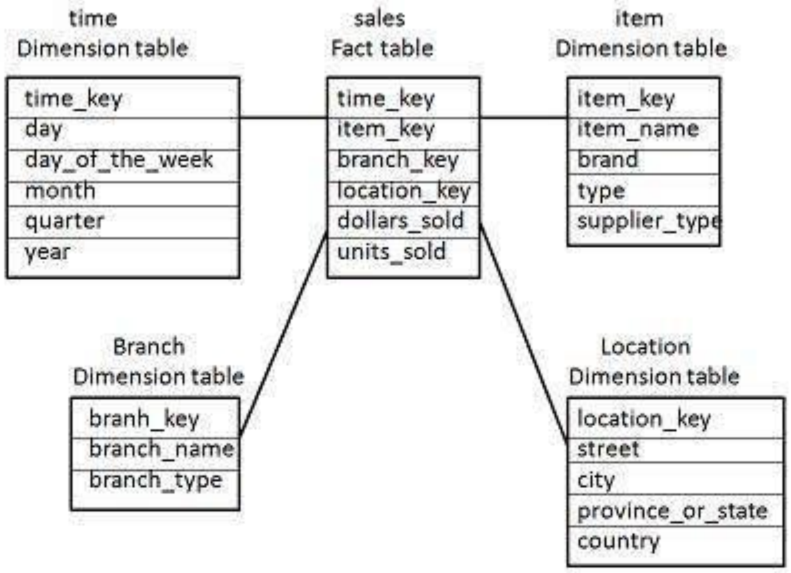
- It presents the data to the user in a form they understand.
- It schedules the execution of the queries posted by the end-user.
- It stores query profiles to allow the warehouse manager to determine which indexes and aggregations are appropriate.

DATABASE SCHEMA

Schema is a logical description of the entire database. It includes the name and description of records of all record types including all associated data-items and aggregates. Much like a database, a data warehouse also requires to maintain a schema. A database uses relational model, while a data warehouse uses Star, Snowflake, and Fact Constellation schema. In this chapter, we will discuss the schemas used in a data warehouse.

Star Schema

- Each dimension in a star schema is represented with only one-dimension table.
- This dimension table contains the set of attributes.
- The following diagram shows the sales data of a company with respect to the four dimensions, namely time, item, branch, and location.

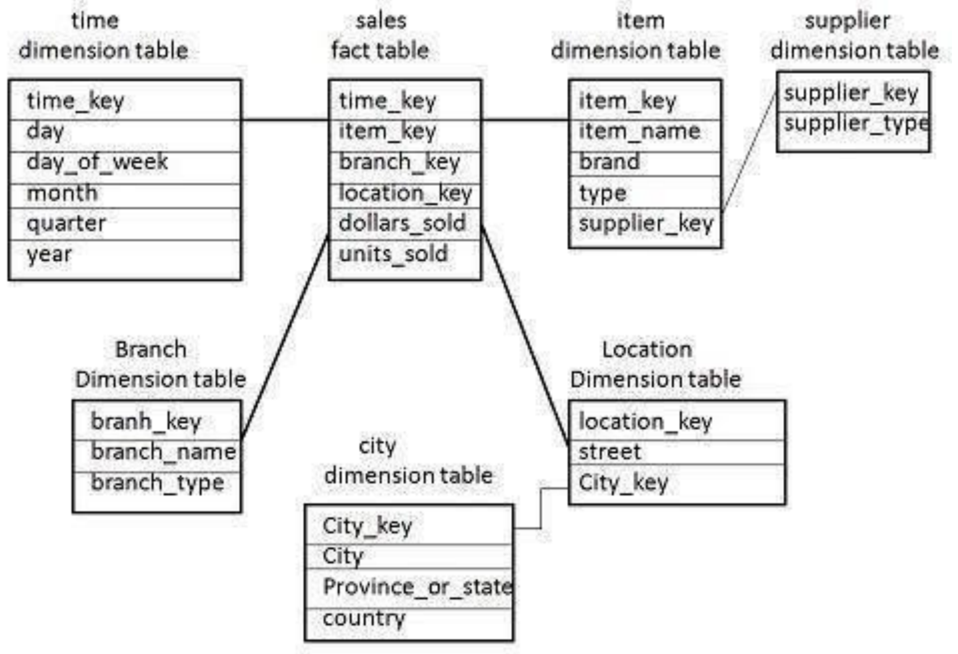


- There is a fact table at the center. It contains the keys to each of four dimensions.
- The fact table also contains the attributes, namely dollars sold and units sold.

Note – Each dimension has only one dimension table and each table holds a set of attributes. For example, the location dimension table contains the attribute set {location_key, street, city, province_or_state,country}. This constraint may cause data redundancy. For example, "Vancouver" and "Victoria" both the cities are in the Canadian province of British Columbia. The entries for such cities may cause data redundancy along the attributes province_or_state and country.

Snowflake Schema

- Some dimension tables in the Snowflake schema are normalized.
- The normalization splits up the data into additional tables.
- Unlike Star schema, the dimensions table in a snowflake schema are normalized. For example, the item dimension table in star schema is normalized and split into two dimension tables, namely item and supplier table.

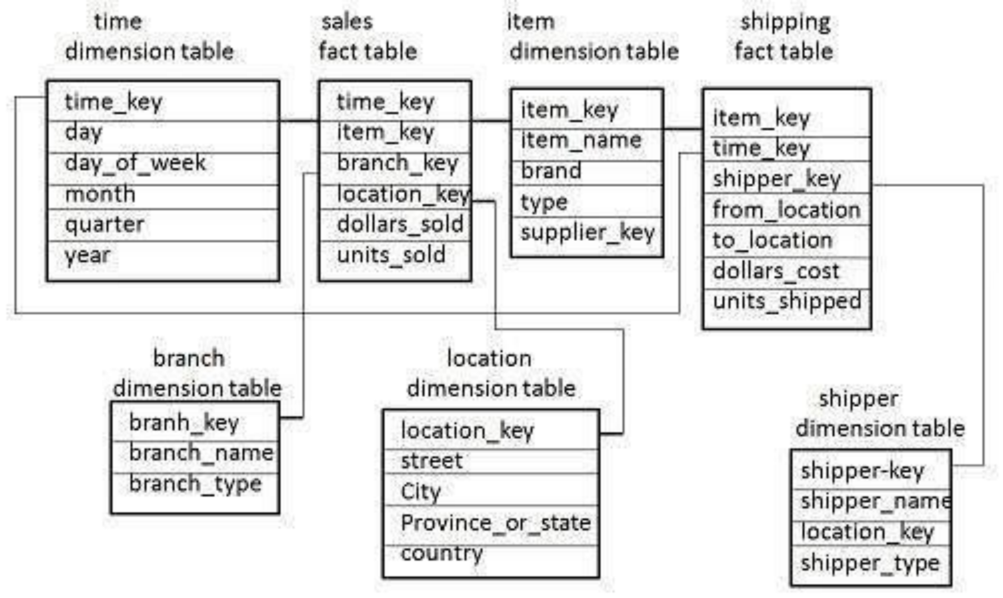


- Now the item dimension table contains the attributes item_key, item_name, type, brand, and supplier-key.
- The supplier key is linked to the supplier dimension table. The supplier dimension table contains the attributes supplier_key and supplier_type.

Note – Due to normalization in the Snowflake schema, the redundancy is reduced and therefore, it becomes easy to maintain and the save storage space.

Fact Constellation Schema

- A fact constellation has multiple fact tables. It is also known as galaxy schema.
- The following diagram shows two fact tables, namely sales and shipping.



- The sales fact table is same as that in the star schema.
- The shipping fact table has the five dimensions, namely item_key, time_key, shipper_key, from_location, to_location.
- The shipping fact table also contains two measures, namely dollars sold and units sold.
- It is also possible to share dimension tables between fact tables. For example, time, item, and location dimension tables are shared between the sales and shipping fact table.

Schema Definition

Multidimensional schema is defined using Data Mining Query Language (DMQL). The two primitives, cube definition and dimension definition, can be used for defining the data warehouses and data marts.

Syntax for Cube Definition

```
define cube < cube_name > [ < dimension-list > }: < measure_list >
```

Syntax for Dimension Definition

```
define dimension < dimension_name > as ( < attribute_or_dimension_list > )
```

Star Schema Definition

The star schema that we have discussed can be defined using Data Mining Query Language (DMQL) as follows −

```
define cube sales star [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state, country)
```

Snowflake Schema Definition

Snowflake schema can be defined using DMQL as follows −

```
define cube sales snowflake [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier (supplier key, supplier type))
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city (city key, city, province or state, country))
```

Fact Constellation Schema Definition

Fact constellation schema can be defined using DMQL as follows −

```
define cube sales [time, item, branch, location]:

dollars sold = sum(sales in dollars), units sold = count(*)

define dimension time as (time key, day, day of week, month, quarter, year)
define dimension item as (item key, item name, brand, type, supplier type)
define dimension branch as (branch key, branch name, branch type)
define dimension location as (location key, street, city, province or state,country)
define cube shipping [time, item, shipper, from location, to location]:
```

```
dollars cost = sum(cost in dollars), units shipped = count(*)

define dimension time as time in cube sales
define dimension item as item in cube sales
define dimension shipper as (shipper key, shipper name, location as location in cube sales, shipper type)
define dimension from location as location in cube sales
define dimension to location as location in cube sales
```

PARTITION STRATEGY

Partitioning is done to enhance performance and facilitate easy management of data. Partitioning also helps in balancing the various requirements of the system. It optimizes the hardware performance and simplifies the management of data warehouse by partitioning each fact table into multiple separate partitions. In this chapter, we will discuss different partitioning strategies.

Why is it Necessary to Partition?

Partitioning is important for the following reasons −

- For easy management,
- To assist backup/recovery,
- To enhance performance.

For Easy Management

The fact table in a data warehouse can grow up to hundreds of gigabytes in size. This huge size of fact table is very hard to manage as a single entity. Therefore it needs partitioning.

To Assist Backup/Recovery

If we do not partition the fact table, then we have to load the complete fact table with all the data. Partitioning allows us to load only as much data as is required on a regular basis. It reduces the time to load and also enhances the performance of the system.

Note − To cut down on the backup size, all partitions other than the current partition can be marked as read-only. We can then put these partitions into a state where they cannot be modified. Then they can be backed up. It means only the current partition is to be backed up.

To Enhance Performance

By partitioning the fact table into sets of data, the query procedures can be enhanced. Query performance is enhanced because now the query scans only those partitions that are relevant. It does not have to scan the whole data.

Horizontal Partitioning

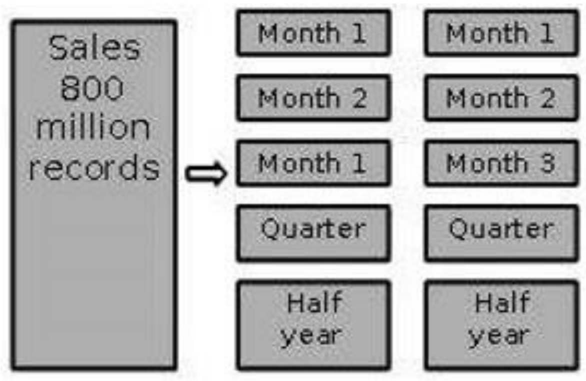
There are various ways in which a fact table can be partitioned. In horizontal partitioning, we have to keep in mind the requirements for manageability of the data warehouse.

Partitioning by Time into Equal Segments

In this partitioning strategy, the fact table is partitioned on the basis of time period. Here each time period represents a significant retention period within the business. For example, if the user queries for **month to date data** then it is appropriate to partition the data into monthly segments. We can reuse the partitioned tables by removing the data in them.

Partition by Time into Different-sized Segments

This kind of partition is done where the aged data is accessed infrequently. It is implemented as a set of small partitions for relatively current data, larger partition for inactive data.



Points to Note

- The detailed information remains available online.
- The number of physical tables is kept relatively small, which reduces the operating cost.
- This technique is suitable where a mix of data dipping recent history and data mining through entire history is required.
- This technique is not useful where the partitioning profile changes on a regular basis, because repartitioning will increase the operation cost of data warehouse.

Partition on a Different Dimension

The fact table can also be partitioned on the basis of dimensions other than time such as product group, region, supplier, or any other dimension. Let's have an example.

Suppose a market function has been structured into distinct regional departments like on a **state by state** basis. If each region wants to query on information captured within its region, it would prove to be more effective to partition the fact table into regional partitions. This will cause the queries to speed up because it does not require to scan information that is not relevant.

Points to Note

- The query does not have to scan irrelevant data which speeds up the query process.
- This technique is not appropriate where the dimensions are unlikely to change in future. So, it is worth determining that the dimension does not change in future.
- If the dimension changes, then the entire fact table would have to be repartitioned.

Note − We recommend to perform the partition only on the basis of time dimension, unless you are certain that the suggested dimension grouping will not change within the life of the data warehouse.

Partition by Size of Table

When there are no clear basis for partitioning the fact table on any dimension, then we should **partition the fact table on the basis of their size.** We can set the predetermined size as a critical point. When the table exceeds the predetermined size, a new table partition is created.

Points to Note

- This partitioning is complex to manage.
- It requires metadata to identify what data is stored in each partition.

Partitioning Dimensions

If a dimension contains large number of entries, then it is required to partition the dimensions. Here we have to check the size of a dimension.

Consider a large design that changes over time. If we need to store all the variations in order to apply comparisons, that dimension may be very large. This would definitely affect the response time.

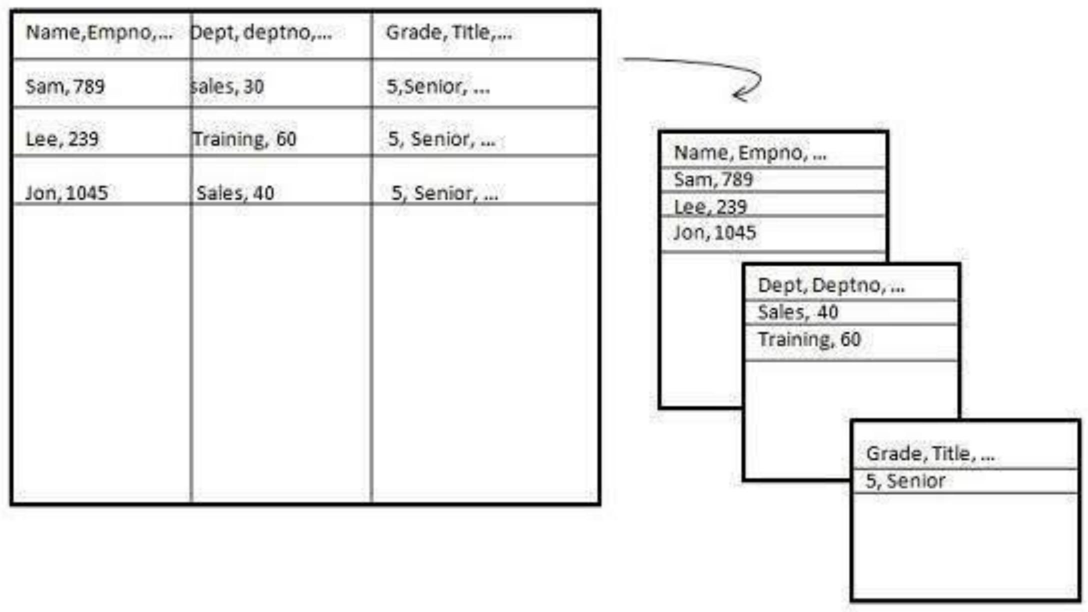
Round Robin Partitions

In the round robin technique, when a new partition is needed, the old one is archived. It uses metadata to allow user access tool to refer to the correct table partition.

This technique makes it easy to automate table management facilities within the data warehouse.

Vertical Partition

Vertical partitioning, splits the data vertically. The following images depicts how vertical partitioning is done.



Vertical partitioning can be performed in the following two ways −

- Normalization
- Row Splitting

Normalization

Normalization is the standard relational method of database organization. In this method, the rows are collapsed into a single row, hence it reduce space. Take a look at the following tables that show how normalization is performed.

Table before Normalization

Product_id	Qty	Value	sales_date	Store_id	Store_name	Location	Region

30	5	3.67	3-Aug-13	16	sunny	Bangalore	S
35	4	5.33	3-Sep-13	16	sunny	Bangalore	S
40	5	2.50	3-Sep-13	64	San	Mumbai	W
45	7	5.66	3-Sep-13	16	sunny	Bangalore	S

Table after Normalization

Store_id	**Store_name**	**Location**	**Region**
16	sunny	Bangalore	W
64	san	Mumbai	S

Product_id	**Quantity**	**Value**	**sales_date**	**Store_id**
30	5	3.67	3-Aug-13	16
35	4	5.33	3-Sep-13	16
40	5	2.50	3-Sep-13	64
45	7	5.66	3-Sep-13	16

Row Splitting

Row splitting tends to leave a one-to-one map between partitions. The motive of row splitting is to speed up the access to large table by reducing its size.

Note − While using vertical partitioning, make sure that there is no requirement to perform a major join operation between two partitions.

Identify Key to Partition

It is very crucial to choose the right partition key. Choosing a wrong partition key will lead to reorganizing the fact table. Let's have an example. Suppose we want to partition the following table.

Account_Txn_Table

```
transaction_id
account_id
transaction_type
value
transaction_date
region
branch_name
```

We can choose to partition on any key. The two possible keys could be

- region
- transaction_date

Suppose the business is organized in 30 geographical regions and each region has different number of branches. That will give us 30 partitions, which is reasonable. This partitioning is good enough because our requirements capture has shown that a vast majority of queries are restricted to the user's own business region.

If we partition by transaction_date instead of region, then the latest transaction from every region will be in one partition. Now the user who wants to look at data within his own region has to query across multiple partitions.

Hence it is worth determining the right partitioning key.

<u>AGGREATIONS</u>

WHY AGGREGATE?

Data aggregation is an essential component of decision support data warehouses. It allows us top provide cost-effective query performance, by avoiding the need for substantial investment in processing power in order to address performance Having said that, it is important to note that there is a fine line to walk when requirements. designing an aggregation strategy. Too many aggregations will lead to unacceptable operational costs; too few will lead to an overall lack of system performance. advise that the best that can be hoped for amount of is a data warehouse solution where 70% of the queries respond in a reasonable As a guideline, we often time. The rest will take substantially longer unless the business is willing to make additional, possibly excessive, investments in hardware processing power.

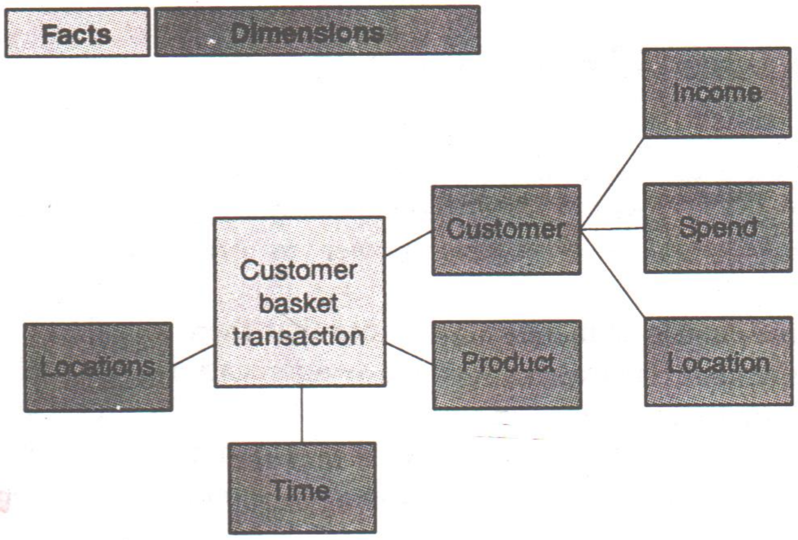
In addition, aggregations allow us to spot trends within the data that may be difficult to see by looking at the detail. The old adage of "not being able to see the wood for trees" is particularly appropriate here.

WHAT IS AN AGGREGATION?

Aggregation strategies rely on the fact that most common queries will analyze either a subset or an aggregation of the detailed data. A large number of queries in a typical decision support data warehouse will ask questions that analyze data dimensionally. In most instances, you will find that those queries will examine detailed data by either a subset or an aggregation of one or more dimensions. This will be driven by the business need, rather than by the perceived information need.

For example, let us consider a situation where a marketing executive within a supermarket chain wishes to encourage the sales of lamb products in a geographic region he or she are responsible for. Part of this process is likely to involve identifying existing customers who are likely to increase their individual lamb purchases. This could be achieved by offering discounts on existing lamb products, or through special promotions that would encourage customers to switch to lamb from other meat products.

The marketing executive should be able to analyze the customer profiles within a district, and determine the income, spend, and eating preference for customers within that district. It could be rather insensitive for the supermarket to promote



Lamp sales to customers who have ethical or religious issues with the consumption of

In this instance, a possible query would examine:

income,

spend,

a district in question aggregated by sub districts — for example, postal regions within Orange County, purchasing trend over a reasonable period of time (for example, 2 years),

purchasing trend for meat buyers, basket all existing loyalty customers who have made purchases within the past month. We can see that the query is analyzing customer basket data by five dimensions: come, spend, location, time, and product. At this point, a number of facts are apparent:

• Sub district is an aggregation of location.

• District is a subset of location.

• The time period is substantial: that is, it will result in the analysis of the bulk of the detailed basket data.

• Meat buyers are customers who purchase a specific subset of the product dimension.

• Income and spend are dimensions of customer, not customer transaction.

• The nature of the query is such that the locations of interest are customer addresses, not store locations.

All these points would lead us to deduce that this query would need to perform the following operations in sequence:

I Identify customers within Orange County.

2 Identify the subset of customers who purchase meat regularly, by scanning purchases over the past two years.

3 Identify income and spend for each customer.

This all sounds rather complicated and time consuming, and would be greatly simplified by having a pre-aggregated summary 'table that identifies all meat-buying customers in advance This does not mean to say that the query is changed in any way, but the bulk of the processing is performed in advance of the query, and can be used repetitively, for example to analyze any other trends about this type of customers We can see that the appropriate aggregation will substantially reduce the processing time required to run a query, at the cost of preprocessing and storing the intermediate results.

We can also see within the example that the advantages of pre-aggregating meat-buying customers could change at short notice, once the business users focus on a different set of conditions.

DESIGNING SUMMARY TABLES

As we discussed earlier, the primary purpose of using summary tables is to cut down time it takes to execute queries. As a direct consequence of this, the design Orgy for summary tables is substantially different from that for fact tables.

In essence, the overriding objective when using a summary table is to minimize volume of data being scanned, by storing as many partial results as possible. This tot just about storing aggregated values in columns; it may for example be ropriate to embed reference data into the summary table in order to cut down the r it takes to execute joins.

The following points differ from fact table design when designing summary

- use of intelligent or non-intelligent keys;
- use of abbreviations decoded on the fly;
- use of embedded reference keys;
- which columns to include — that is, vertical partitioning strategy;
- horizontal partitioning strategy.

Summary tables are designed using a similar process to that followed to design fact les, amended to take into account the differences in strategy. The steps in the process are:

- Determine which dimensions are aggregated.
- Determine the aggregation of multiple values.
- Aggregate multiple facts into the summary table.
- Determine the level of aggregation.
- Determine the extent of embedding dimension data in the summary.
- Design time into the summary table.
- Index the summary table.

DATA MARTING

Why Do We Need a Data Mart?

Listed below are the reasons to create a data mart −

- To partition data in order to impose **access control strategies.**
- To speed up the queries by reducing the volume of data to be scanned.
- To segment data into different hardware platforms.
- To structure data in a form suitable for a user access tool.

Note − Do not data mart for any other reason since the operation cost of data marting could be very high. Before data marting, make sure that data marting strategy is appropriate for your particular solution.

Cost-effective Data Marting

Follow the steps given below to make data marting cost-effective −

- Identify the Functional Splits
- Identify User Access Tool Requirements
- Identify Access Control Issues

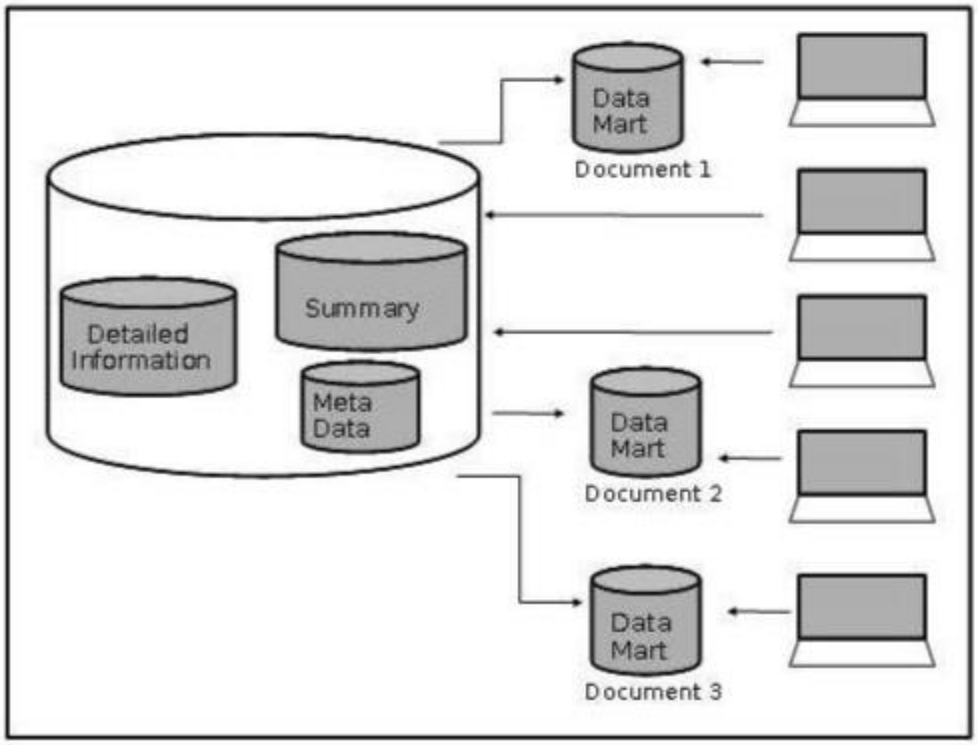
Identify the Functional Splits

In this step, we determine if the organization has natural functional splits. We look for departmental splits, and we determine whether the way in which departments use information tend to be in isolation from the rest of the organization. Let's have an example.

Consider a retail organization, where each merchant is accountable for maximizing the sales of a group of products. For this, the following are the valuable information −

- sales transaction on a daily basis
- sales forecast on a weekly basis
- stock position on a daily basis
- stock movements on a daily basis

As the merchant is not interested in the products they are not dealing with, the data marting is a subset of the data dealing which the product group of interest. The following diagram shows data marting for different users.



Given below are the issues to be taken into account while determining the functional split −

- The structure of the department may change.
- The products might switch from one department to other.
- The merchant could query the sales trend of other products to analyze what is happening to the sales.

Note − We need to determine the business benefits and technical feasibility of using a data mart.

Identify User Access Tool Requirements

We need data marts to support **user access tools** that require internal data structures. The data in such structures are outside the control of data warehouse but need to be populated and updated on a regular basis.

There are some tools that populate directly from the source system but some cannot. Therefore additional requirements outside the scope of the tool are needed to be identified for future.

Note − In order to ensure consistency of data across all access tools, the data should not be directly populated from the data warehouse, rather each tool must have its own data mart.

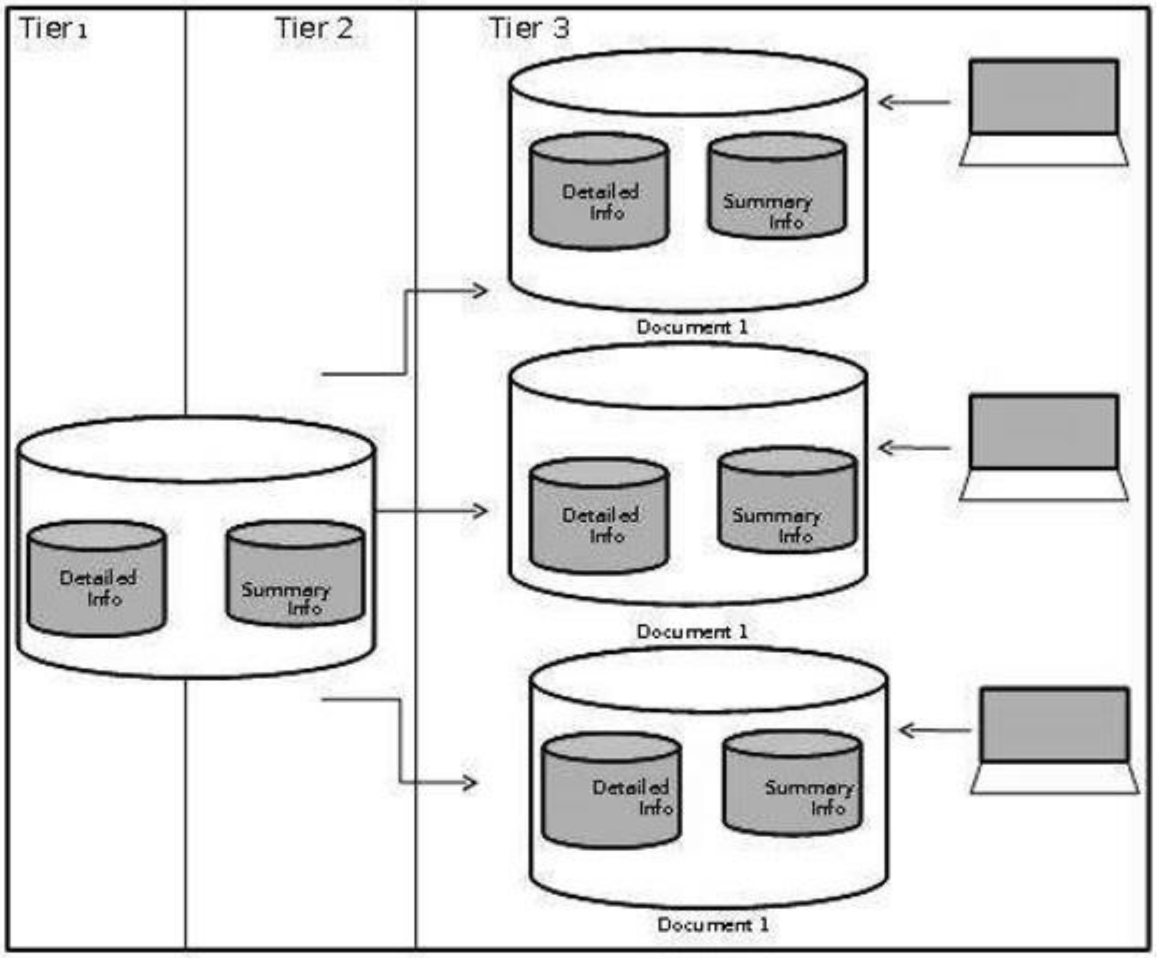
Identify Access Control Issues

There should to be privacy rules to ensure the data is accessed by authorized users only. For example a data warehouse for retail banking institution ensures that all the accounts belong to the same legal entity. Privacy laws can force you to totally prevent access to information that is not owned by the specific bank.

Data marts allow us to build a complete wall by physically separating data segments within the data warehouse. To avoid possible privacy problems, the detailed data can be removed from the data warehouse. We can create data mart for each legal entity and load it via data warehouse, with detailed account data.

Designing Data Marts

Data marts should be designed as a smaller version of starflake schema within the data warehouse and should match with the database design of the data warehouse. It helps in maintaining control over database instances.



The summaries are data marted in the same way as they would have been designed within the data warehouse. Summary tables help to utilize all dimension data in the starflake schema.

Cost of Data Marting

The cost measures for data marting are as follows −

- Hardware and Software Cost
- Network Access
- Time Window Constraints

Hardware and Software Cost

Although data marts are created on the same hardware, they require some additional hardware and software. To handle user queries, it requires additional processing power and disk storage. If detailed data and the data mart exist within the data warehouse, then we would face additional cost to store and manage replicated data.

Note − Data marting is more expensive than aggregations, therefore it should be used as an additional strategy and not as an alternative strategy.

Network Access

A data mart could be on a different location from the data warehouse, so we should ensure that the LAN or WAN has the capacity to handle the data volumes being transferred within the **data mart load process.**

Time Window Constraints

The extent to which a data mart loading process will eat into the available time window depends on the complexity of the transformations and the data volumes being shipped. The determination of how many data marts are possible depends on −

- Network capacity.
- Time window available
- Volume of data being transferred
- Mechanisms being used to insert data into a data mart

META DATA

What is Metadata?

Metadata is simply defined as data about data. The data that is used to represent other data is known as metadata. For example, the index of a book serves as a metadata for the contents in the book. In other words, we can say that metadata is the summarized data that leads us to detailed data. In terms of data warehouse, we can define metadata as follows.

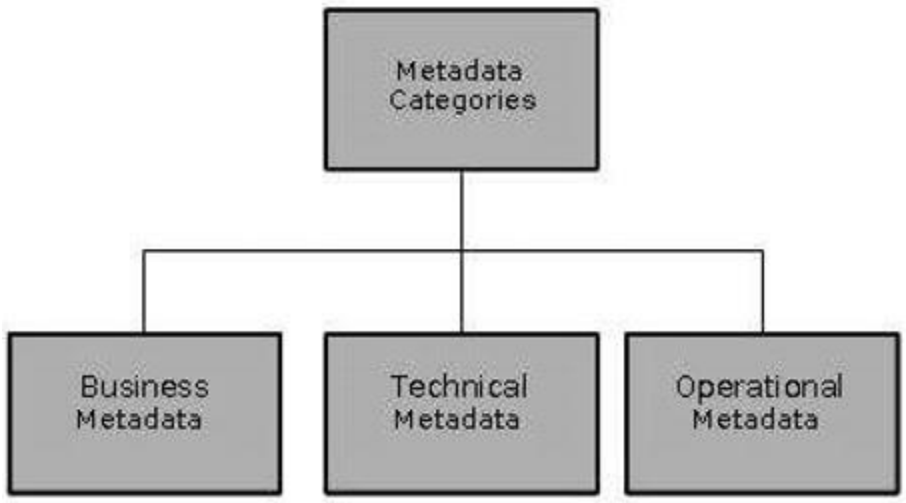
- Metadata is the road-map to a data warehouse.
- Metadata in a data warehouse defines the warehouse objects.
- Metadata acts as a directory. This directory helps the decision support system to locate the contents of a data warehouse.

Note − In a data warehouse, we create metadata for the data names and definitions of a given data warehouse. Along with this metadata, additional metadata is also created for time-stamping any extracted data, the source of extracted data.

Categories of Metadata

Metadata can be broadly categorized into three categories −

- **Business Metadata** − It has the data ownership information, business definition, and changing policies.
- **Technical Metadata** − It includes database system names, table and column names and sizes, data types and allowed values. Technical metadata also includes structural information such as primary and foreign key attributes and indices.
- **Operational Metadata** − It includes currency of data and data lineage. Currency of data means whether the data is active, archived, or purged. Lineage of data means the history of data migrated and transformation applied on it.

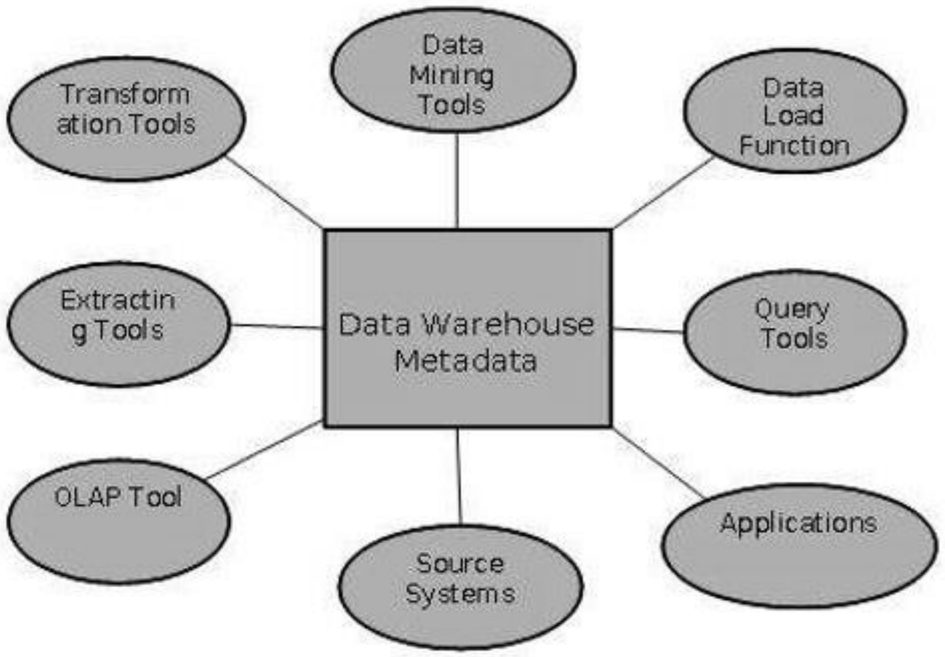


Role of Metadata

Metadata has a very important role in a data warehouse. The role of metadata in a warehouse is different from the warehouse data, yet it plays an important role. The various roles of metadata are explained below.

- Metadata acts as a directory.
- This directory helps the decision support system to locate the contents of the data warehouse.
- Metadata helps in decision support system for mapping of data when data is transformed from operational environment to data warehouse environment.
- Metadata helps in summarization between current detailed data and highly summarized data.
- Metadata also helps in summarization between lightly detailed data and highly summarized data.
- Metadata is used for query tools.
- Metadata is used in extraction and cleansing tools.
- Metadata is used in reporting tools.
- Metadata is used in transformation tools.
- Metadata plays an important role in loading functions.

The following diagram shows the roles of metadata.



Metadata Repository

Metadata repository is an integral part of a data warehouse system. It has the following metadata −

- **Definition of data warehouse** − It includes the description of structure of data warehouse. The description is defined by schema, view, hierarchies, derived data definitions, and data mart locations and contents.
- **Business metadata** − It contains has the data ownership information, business definition, and changing policies.
- **Operational Metadata** − It includes currency of data and data lineage. Currency of data means whether the data is active, archived, or purged. Lineage of data means the history of data migrated and transformation applied on it.
- **Data for mapping from operational environment to data warehouse** − It includes the source databases and their contents, data extraction, data partition cleaning, transformation rules, data refresh and purging rules.
- **Algorithms for summarization** − It includes dimension algorithms, data on granularity, aggregation, summarizing, etc.

Challenges for Metadata Management

The importance of metadata can not be overstated. Metadata helps in driving the accuracy of reports, validates data transformation, and ensures the accuracy of calculations. Metadata also enforces the definition of business terms to business end-users. With all these uses of metadata, it also has its challenges. Some of the challenges are discussed below.

- Metadata in a big organization is scattered across the organization. This metadata is spread in spreadsheets, databases, and applications.
- Metadata could be present in text files or multimedia files. To use this data for information management solutions, it has to be correctly defined.
- There are no industry-wide accepted standards. Data management solution vendors have narrow focus.
- There are no easy and accepted methods of passing metadata.

SYSTEM MANAGER

System management is mandatory for the successful implementation of a data warehouse. The most important system managers are −

- System configuration manager
- System scheduling manager
- System event manager
- System database manager
- System backup recovery manager

System Configuration Manager

- The system configuration manager is responsible for the management of the setup and configuration of data warehouse.
- The structure of configuration manager varies from one operating system to another.

- In Unix structure of configuration, the manager varies from vendor to vendor.
- Configuration managers have single user interface.
- The interface of configuration manager allows us to control all aspects of the system.

Note − The most important configuration tool is the I/O manager.

System Scheduling Manager

System Scheduling Manager is responsible for the successful implementation of the data warehouse. Its purpose is to schedule ad hoc queries. Every operating system has its own scheduler with some form of batch control mechanism. The list of features a system scheduling manager must have is as follows −

- Work across cluster or MPP boundaries
- Deal with international time differences
- Handle job failure
- Handle multiple queries
- Support job priorities
- Restart or re-queue the failed jobs
- Notify the user or a process when job is completed
- Maintain the job schedules across system outages
- Re-queue jobs to other queues
- Support the stopping and starting of queues
- Log Queued jobs
- Deal with inter-queue processing

Note − The above list can be used as evaluation parameters for the evaluation of a good scheduler.

Some important jobs that a scheduler must be able to handle are as follows −

- Daily and ad hoc query scheduling
- Execution of regular report requirements

- Data load
- Data processing
- Index creation
- Backup
- Aggregation creation
- Data transformation

Note − If the data warehouse is running on a cluster or MPP architecture, then the system scheduling manager must be capable of running across the architecture.

System Event Manager

The event manager is a kind of a software. The event manager manages the events that are defined on the data warehouse system. We cannot manage the data warehouse manually because the structure of data warehouse is very complex. Therefore we need a tool that automatically handles all the events without any intervention of the user.

Note − The Event manager monitors the events occurrences and deals with them. The event manager also tracks the myriad of things that can go wrong on this complex data warehouse system.

Events

Events are the actions that are generated by the user or the system itself. It may be noted that the event is a measurable, observable, occurrence of a defined action.

Given below is a list of common events that are required to be tracked.

- Hardware failure
- Running out of space on certain key disks
- A process dying
- A process returning an error
- CPU usage exceeding an 805 threshold
- Internal contention on database serialization points

- Buffer cache hit ratios exceeding or failure below threshold
- A table reaching to maximum of its size
- Excessive memory swapping
- A table failing to extend due to lack of space
- Disk exhibiting I/O bottlenecks
- Usage of temporary or sort area reaching a certain thresholds
- Any other database shared memory usage

The most important thing about events is that they should be capable of executing on their own. Event packages define the procedures for the predefined events. The code associated with each event is known as event handler. This code is executed whenever an event occurs.

System and Database Manager

System and database manager may be two separate pieces of software, but they do the same job. The objective of these tools is to automate certain processes and to simplify the execution of others. The criteria for choosing a system and the database manager are as follows −

- increase user's quota.
- assign and de-assign roles to the users
- assign and de-assign the profiles to the users
- perform database space management
- monitor and report on space usage
- tidy up fragmented and unused space
- add and expand the space
- add and remove users
- manage user password
- manage summary or temporary tables
- assign or deassign temporary space to and from the user
- reclaim the space form old or out-of-date temporary tables
- manage error and trace logs

- to browse log and trace files
- redirect error or trace information
- switch on and off error and trace logging
- perform system space management
- monitor and report on space usage
- clean up old and unused file directories
- add or expand space.

System Backup Recovery Manager

The backup and recovery tool makes it easy for operations and management staff to back-up the data. Note that the system backup manager must be integrated with the schedule manager software being used. The important features that are required for the management of backups are as follows −

- Scheduling
- Backup data tracking
- Database awareness

Backups are taken only to protect against data loss. Following are the important points to remember −

- The backup software will keep some form of database of where and when the piece of data was backed up.
- The backup recovery manager must have a good front-end to that database.
- The backup recovery software should be database aware.
- Being aware of the database, the software then can be addressed in database terms, and will not perform backups that would not be viable.

PROCESS MANAGER

Process managers are responsible for maintaining the flow of data both into and out of the data warehouse. There are three different types of process managers −

- Load manager
- Warehouse manager

- Query manager

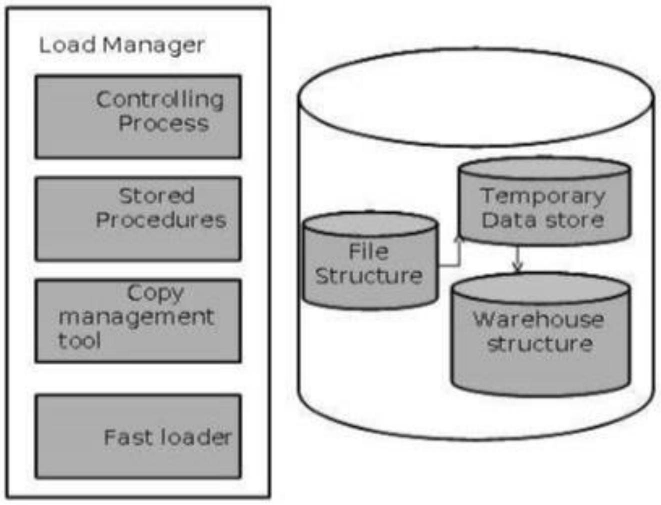
Data Warehouse Load Manager

Load manager performs the operations required to extract and load the data into the database. The size and complexity of a load manager varies between specific solutions from one data warehouse to another.

Load Manager Architecture

The load manager does performs the following functions −

- Extract data from the source system.
- Fast load the extracted data into temporary data store.
- Perform simple transformations into structure similar to the one in the data warehouse.



Extract Data from Source

The data is extracted from the operational databases or the external information providers. Gateways are the application programs that are used to extract data. It is supported by underlying DBMS and allows the client program to generate SQL to be executed at a server. Open Database Connection (ODBC) and Java Database Connection (JDBC) are examples of gateway.

Fast Load

- In order to minimize the total load window, the data needs to be loaded into the warehouse in the fastest possible time.

- Transformations affect the speed of data processing.
- It is more effective to load the data into a relational database prior to applying transformations and checks.
- Gateway technology is not suitable, since they are inefficient when large data volumes are involved.

Simple Transformations

While loading, it may be required to perform simple transformations. After completing simple transformations, we can do complex checks. Suppose we are loading the EPOS sales transaction, we need to perform the following checks −

- Strip out all the columns that are not required within the warehouse.
- Convert all the values to required data types.

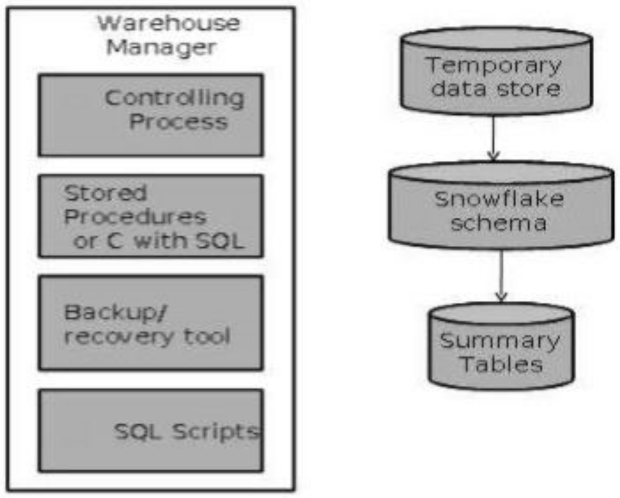
Warehouse Manager

The warehouse manager is responsible for the warehouse management process. It consists of a third-party system software, C programs, and shell scripts. The size and complexity of a warehouse manager varies between specific solutions.

Warehouse Manager Architecture

A warehouse manager includes the following −

- The controlling process
- Stored procedures or C with SQL
- Backup/Recovery tool
- SQL scripts



Functions of Warehouse Manager

A warehouse manager performs the following functions −

- Analyzes the data to perform consistency and referential integrity checks.
- Creates indexes, business views, partition views against the base data.
- Generates new aggregations and updates the existing aggregations.
- Generates normalizations.
- Transforms and merges the source data of the temporary store into the published data warehouse.
- Backs up the data in the data warehouse.
- Archives the data that has reached the end of its captured life.

Note − A warehouse Manager analyzes query profiles to determine whether the index and aggregations are appropriate.

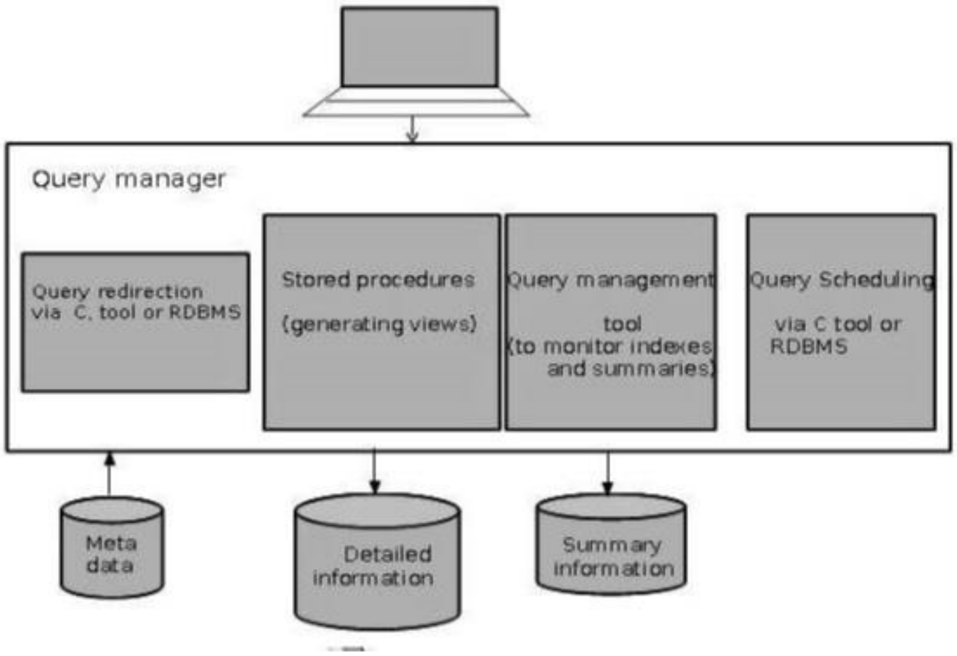
Query Manager

The query manager is responsible for directing the queries to suitable tables. By directing the queries to appropriate tables, it speeds up the query request and response process. In addition, the query manager is responsible for scheduling the execution of the queries posted by the user.

Query Manager Architecture

A query manager includes the following components −

- Query redirection via C tool or RDBMS
- Stored procedures
- Query management tool
- Query scheduling via C tool or RDBMS
- Query scheduling via third-party software


Query manager
Query redirection via C, tool or RDBMS
Stored procedures (generating views)
Query management tool (to monitor indexes and summaries)
Query Scheduling via C tool or RDBMS
Meta data
Detailed information
Summary information

Functions of Query Manager

- It presents the data to the user in a form they understand.
- It schedules the execution of the queries posted by the end-user.
- It stores query profiles to allow the warehouse manager to determine which indexes and aggregations are appropriate.

UNIT – IV Hardware and Operational Design

Hardware Architecture

INTRODUCTION

The aim of this chapter is to cover the issues raised by the choice of hardware. Any data warehouse solution will contain many different pieces of hardware, ranging from the database server to the network and the user front-end hardware. This chapter deals primarily with the possible server environments, covering them in some depth. Network and client-side issues are generally outside the scope of the data warehouse design team. However, there are still some network and client-side issues that need to be considered, and these are covered below.

PROCESS

The hardware architecture of a data warehouse is defined within the technical blueprint stage of the process (Figure 11.1). The business requirements stage should have identified the initial user requirements and given an indication of the capacity-planning requirements. The hardware architecture is determined once a broad understanding of the required technical architecture has been achieved. The backup and security strategies are also determined during the technical blueprint phase.

SERVER HARDWARE

The server is a crucial part of the data warehouse environment. To support the size of the database, and the ad hoc nature of the query access, warehouse applications generally require large hardware configurations. There are a number of different hardware architectures in the open systems market, each of which has its own advantages and disadvantages. The different architectures are discussed below.

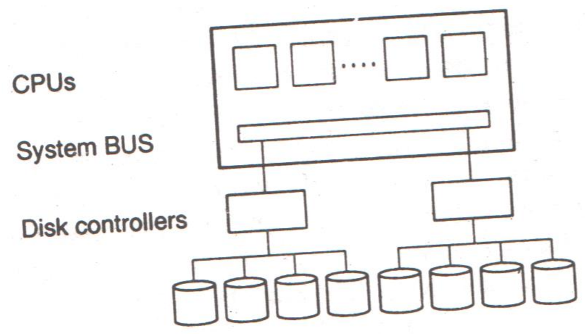
ARCHITECTURE OPTIONS

There are two main hardware architectures commonly used as server platforms in data warehouse solutions: symmetric multi-processing (SMP), and massively parallel processing (MPP). There is a lot of confusion about the distinction between these architectures, and this is not helped by the existence of hybrid machines that use both. The primary distinguishing feature between SMP and MPP is as follows. An SMP machine is a set of tightly coupled CPUs that share memory and disk. An MPP machine is a set of loosely coupled CPUs, each of which has its own memory and disk. Bear this in mind as we discuss each of the architectures in detail below. To add to the confusion, new and emerging technologies, such as extremely high-speed memory connections and non-uniform memory architecture (NUMA), are adding variants of the SMP architecture, allowing them to scale to much higher levels.

SYMMETRIC MULTI-PROCESSING

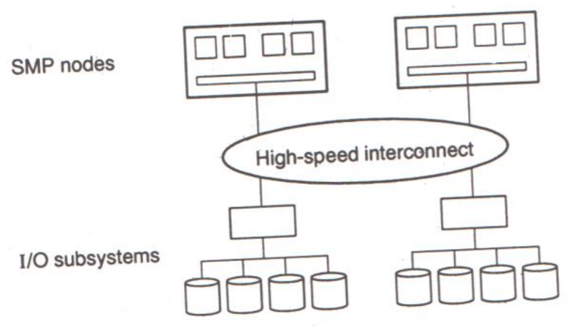
Illustrates a typical SMP configuration. SMP machines are the natural ,Progression of the traditional single-CPU midrange servers. SMP, along with the growth of clustering technology, has made possible the support of larger and larger databases. They are also the base technologies that enable the downsizing of rainframe systems. SMP gives the CPU bandwidth needed to support the large ad hoc queries of decision support systems.

An SMP machine is a set of CPUs that share memory and disk. This is times called a shared-everything environment. Unlike some of the earlier multi-CPU machines, where there existed a master CPU with a number of slave CPUs, the CPUs in an SMP machine are all equal. The operating system. contains special code k allow the different CPUs to access central structures, such as memory address tables and job schedule queues, in a controlled manner. This means that a process can run on any CPU in the machine, and indeed it will often be run on different CPUs at different times.



CLUSTER TECHNOLOGY

An Illustrates a typical cluster system. A cluster is a set of loosely coupled SMP machines connected by a high-speed interconnect. Each machine has its own CPUs and memory, but they share access to disk. Thus these systems are called shared-disk systems. Each machine in the cluster is called a node. The aim of the cluster is to mimic a single larger machine. In this pseudo single machine, resources such as shared disk must be managed in a distributed fashion. A layer of software called the distributed lock manager is used to achieve this. This software is run on each machine, and communicates over the high-speed interconnect. The distributed lock manager maintains the integrity of each resource by ensuring that the resource can be changed on only one node of the cluster at a time.



MASSIVELY PARALLEL PROCESSING

AN Illustrates a typical MPP system. Massively parallel processing is the "new kid on the block." Like any new technology, MPP has had its teething problems, not least of which is the lack of tools to manage, monitor and support these massive machines. This is starting to change as the MPP market shakes itself out.

Most of the myriad MPP start-up companies have failed, leaving only a few serious players in the MPP business. However, MPP has for the most part failed to get a toehold in the IT strategies of general business. This is likely to be compounded by the advent of the new SMP architectures that are emerging, which allow much greater SMP scalability. So, for the foreseeable future, MPP is likely to be concentrated in certain niche markets.

An MPP machine is made up of many loosely coupled nodes. These nodes will be linked together by a high-speed connection. The form that this connection takes varies from vendor to vendor. Each node has its own memory, and the disks are not shared, each being attached to only one node. However, most MPP systems allow a disk to be dual connected between two nodes. This protects against an individual node failure causing disks to be unavailable.

NEW AND EMERGING TECHNOLOGIES

Non uniform memory architecture (NUMA) machines are not a new concept. The idea has been around for some years. Recent advances in hardware technology have made machines such as the Sequent NUMA-Q a possibility. A NUMA machine is basically a tightly coupled cluster of SMP nodes, with an extremely high-speed interconnect. The interconnect needs to be sufficiently fast to give near-SMP internal speeds. The advantage of such an interconnect is that it enables the operating system to run across the whole machine as a single distributed instance, rather than the usual separate copies of the operating system running on each node. In essence it makes the distributed memory on the SMP nodes of a NUMA machine act as if it were a single shared memory.

SERVER MANAGEMENT

The systems required for data warehouse environments are generally large and complex. This, added to the changing nature of a warehouse, means that these systems require a lot of management. For systems and database administrators to manage effectively, they require the use of one or more of the increasing number of management and monitoring tools on the market. These tools allow the automatic monitoring of most if not all the required processes and statistics. Events such as:

• running out of space on certain key disks, • a process dying, • a process using excessive resource (such as CPU), • a process returning an error, • disks exhibiting I/O bottlenecks, • hardware failure, • a table failing to extend because of lack of space, • CPU usage exceeding an 80% threshold, • excessive memory swapping, • low buffer cache hit ratios,

NETWORK HARDWARE

The network, although not part of the data warehouse itself, can play an important part in a data warehouse's success.

NETWORK ARCHITECTURE

As long as the network has sufficient bandwidth to supply the data feed and user requirements, the architecture of the network is irrelevant. It is, however, important at the design stage to consider some aspects of the network architecture. For example, if user access is going to be via a wide area network (WAN), this needs to be considered in the design of parts of the application, such as the query manager.

The main aspects of a data warehouse design network architecture are:

• user access, • source system data transfer, • data extractions.

NETWORK MANAGEMENT

Network management is a black art. It requires specialist tools and lots of network experience. The management of the network has no direct effect on a data warehouse, except for one issue. It is important to be able to monitor network performance. The network may play a key part in data flow through a data warehouse environment. Being able to monitor this part of the data flow is necessary to enable resolution of any performance problems.

CLIENT HARDWARE

As with the network, clients are external to the data warehouse system itself. There are, however, still aspects that need to be considered during the design phase.

CLIENT MANAGEMENT

Management of the clients is beyond the scope of the data warehouse environment. The dependencies here are the other way around. Thcse responsible for client machine management will need to know the requirements for that machine to access the data warehouse system. Details such as the network protocols supported on the server, and the server's Internet address, will need to be supplied. If multiple access paths to the server system exist, this information needs to be relayed to those responsible for the client systems. For example, if the server is a cluster environment, during node fall over users may need to access a different machine address. All such dependencies need to be driven out and documented.

CLIENT TOOLS

At the design stage it may be necessary to consider what user-side tools will be used. If these tools have special requirements, such as data being summarized in certain ways, these requirements need to be catered for. Even given this requirement, it is important that the data warehouse be designed to be accessible to any tool. In fact the tool should not be allowed to affect the basic design of the warehouse itself. This protects against changing tools requirements, and is particularly important in the data warehouse arena, where requirements evolve over time.

PHYSICAL LAYOUT

PARALLEL TECHNOLOGY

At the heart of any computer is the central processing unit (CPU). Notwithstanding I 0 bottlenecks and bus and backplane capacities, the speed of the CPU is what dictates the amount of work that a computer can perform in a given time. Many techniques, such as pipelining instructions or the ability to issue multiple instructions simultaneously, are employed to try get around the limits of a single CPU. Another :.-technique that became commercially available in the late 1980s and early 1990s was to make operating system software sophisticated enough to run across multiple CPUs at the same time. This allowed computers to scale to much greater levels of performance.

Clearly, having multiple CPUs in the same machine allows more than one job to be dealt with simultaneously. However, it also opens up another possibility: that of balling a single job running on more than one CPU at the same time. The importance ti this possibility cannot be stressed enough.

PARALLEL QUERY

What is parallel query?

What exactly does it mean to parallelize a query? Before we can answer this question we need to be clear on exactly what we mean by a query. A query is a single statement that requests a certain subset of data ordered or presented in a particular way. This query will be presented to the relational database management system (RDBMS) in a language such as Structured Query Language (SQL). For example, the query

select customer_name, region, surn(amount_paid) from sales where sum (amount_paid) > 1000 and date_of_sale between 101-JAN-96' and 131- JAN-96' group by region customer_name order by region

will bring back the sales figures by region and by customer name, for all customers that have spent more than $1000 in the month of January. The where clause specifies the subset of data required from the sales table. The group by a-id order by clauses specify the way we want the data aggregated and presented. The steps taken by the RDBMS to satisfy the above query are:

1 Fetch the data from the sales table.

2 Apply the restrictions from the where clause.

3 Sort the returned data.

4 Pass sorted data to requesting tool/process.

DISK TECHNOLOGY

RAID TECHNOLOGY

what is RAID? Why do you need to use it? These are two questions that need to be answered and understood before the physical layout of the database can be underertaken. RAID stands for **redundant array of inexpensive disks,** and is a technology that was originally introduced in the paper "A case for redundant arrays expensive disks" by Patterson, Gibson and Katz, of the University of California, limieley, in 1987. The purpose of RAID technology is to provide resilience against disk failure, so that the loss of an individual disk does not mean loss of data. Table defines what each level of RAID means.

Striping is a technique in which data is spread across multiple disks. This allows it higher I/O performance by using the speed of several disks to satisfy a given I/O Bequest. Note that the distinction in Table 12.1 between RAID level 1 and RAID level 0 + 1 is not really required. Striping at RAID level 1 is actually a separate process from the mirroring itself, rather than an integral part of the RAID implementation as it is at other levels. So from here on, when we refer to RAID level ll we mean both RAID level 1 and level 0 + 1.

Parity in this context means that extra data is maintained to provide resilience. :Ln RAID levels 2-4 this is generally achieved by having a separate parity disk that contains the parity information.

DATABASE LAYOUT

Once your logical database design has been completed, it needs to be converted into an actual physical layout, with each data file being mapped to a given disk or set of disks. The mapping actually needs to map a data file to a filesystem or to a raw volume. This is done by use of the system volume manager, which is used to define logical volumes. Each logical volume is an area assigned from a disk or a group of disks. A logical volume can have a filesystem defined on it or can be used as a raw device. There will probably be thousands or even tens of thousands of logical volumes to be defined. This can prove to be a management nightmare.

The key to managing such a huge number of logical volumes is to have a meaningful naming convention. This will probably be limited by the volume manager, as most volume managers only allow short logical volume names. Name lengths as low as 10 are not uncommon. Nonetheless, it is worth putting some effort into producing and documenting a naming convention.

FILE SYSTEMS

FILESYSTEMS VERSUS RAW VOLUMES

One decision that needs to be made by the warehouse designer is whether to use filesystem files or raw devices. First let us define exactly what is meant by filesystems and raw devices.

A filesystem in UNIX is an area of disk set aside for holding files. The filesystem will have a directory structure with a head directory. The filesystem can be mounted into a machine's directory structure by use of the mount command. This allows the filesystem to be mounted anywhere within the system's directory structure. All the normal file and directory commands, such as cd, 1 s, and cat, can be used on the filesystem files.

A raw device is a pseudo device, or area of disk that is treated as a separate named device by UNIX. The raw device can be treated as a single file, which can be read and written directly by a program. The raw device is not subject to all the normal file and directory commands, and does not appear in the system's directory structure, except as a device in the /dev directory. consistent. If the system crashes, a significant amount of

<u>SECURITY</u>

The objective of a data warehouse is to make large amounts of data easily accessible to the users, hence allowing the users to extract information about the business as a whole. But we know that there could be some security restrictions applied on the data that can be an obstacle for accessing the information. If the analyst has a restricted view of data, then it is impossible to capture a complete picture of the trends within the business.

The data from each analyst can be summarized and passed on to management where the different summaries can be aggregated. As the aggregations of summaries cannot be the same as that of the aggregation as a whole, it is possible to miss some information trends in the data unless someone is analyzing the data as a whole.

Security Requirements

Adding security features affect the performance of the data warehouse, therefore it is important to determine the security requirements as early as possible. It is difficult to add security features after the data warehouse has gone live.

During the design phase of the data warehouse, we should keep in mind what data sources may be added later and what would be the impact of adding those data sources. We should consider the following possibilities during the design phase.

- Whether the new data sources will require new security and/or audit restrictions to be implemented?
- Whether the new users added who have restricted access to data that is already generally available?

This situation arises when the future users and the data sources are not well known. In such a situation, we need to use the knowledge of business and the objective of data warehouse to know likely requirements.

The following activities get affected by security measures −

- User access
- Data load
- Data movement
- Query generation

User Access

We need to first classify the data and then classify the users on the basis of the data they can access. In other words, the users are classified according to the data they can access.

Data Classification

The following two approaches can be used to classify the data −

- Data can be classified according to its sensitivity. Highly-sensitive data is classified as highly restricted and less-sensitive data is classified as less restrictive.
- Data can also be classified according to the job function. This restriction allows only specific users to view particular data. Here we restrict the users to view only that part of the data in which they are interested and are responsible for.

There are some issues in the second approach. To understand, let's have an example. Suppose you are building the data warehouse for a bank. Consider that the data being stored in the data warehouse is the transaction data for all the accounts. The question here is, who is allowed to see the transaction data. The solution lies in classifying the data according to the function.

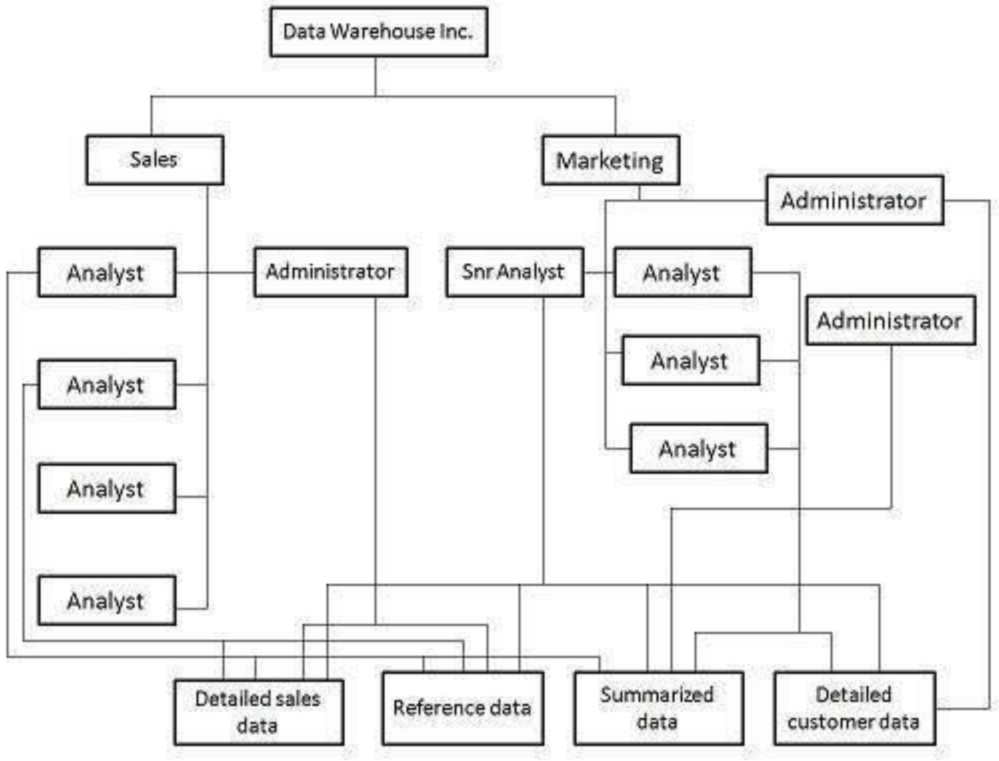
User classification

The following approaches can be used to classify the users –

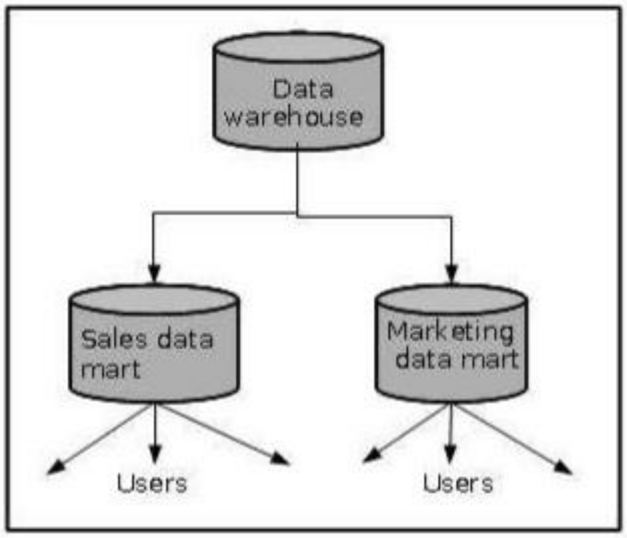
- Users can be classified as per the hierarchy of users in an organization, i.e., users can be classified by departments, sections, groups, and so on.
- Users can also be classified according to their role, with people grouped across departments based on their role.

Classification on basis of Department

Let's have an example of a data warehouse where the users are from sales and marketing department. We can have security by top-to-down company view, with access centered on the different departments. But there could be some restrictions on users at different levels. This structure is shown in the following diagram.

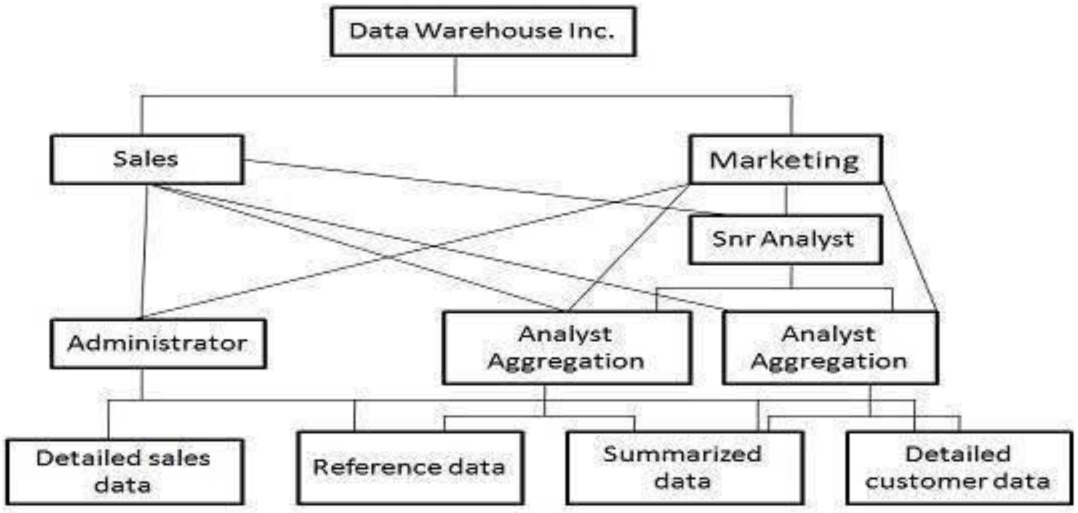


But if each department accesses different data, then we should design the security access for each department separately. This can be achieved by departmental data marts. Since these data marts are separated from the data warehouse, we can enforce separate security restrictions on each data mart. This approach is shown in the following figure.



Classification Based on Role

If the data is generally available to all the departments, then it is useful to follow the role access hierarchy. In other words, if the data is generally accessed by all the departments, then apply security restrictions as per the role of the user. The role access hierarchy is shown in the following figure.



Audit Requirements

Auditing is a subset of security, a costly activity. Auditing can cause heavy overheads on the system. To complete an audit in time, we require more hardware and therefore, it is recommended that wherever possible, auditing should be switched off. Audit requirements can be categorized as follows −

- Connections
- Disconnections
- Data access
- Data change

Note − For each of the above-mentioned categories, it is necessary to audit success, failure, or both. From the perspective of security reasons, the auditing of failures are very important. Auditing of failure is important because they can highlight unauthorized or fraudulent access.

Network Requirements

Network security is as important as other securities. We cannot ignore the network security requirement. We need to consider the following issues −

- Is it necessary to encrypt data before transferring it to the data warehouse?
- Are there restrictions on which network routes the data can take?

These restrictions need to be considered carefully. Following are the points to remember −

- The process of encryption and decryption will increase overheads. It would require more processing power and processing time.
- The cost of encryption can be high if the system is already a loaded system because the encryption is borne by the source system.

Data Movement

There exist potential security implications while moving the data. Suppose we need to transfer some restricted data as a flat file to be loaded. When the data is loaded into the data warehouse, the following questions are raised −

- Where is the flat file stored?
- Who has access to that disk space?

If we talk about the backup of these flat files, the following questions are raised −

- Do you backup encrypted or decrypted versions?
- Do these backups need to be made to special tapes that are stored separately?
- Who has access to these tapes?

Some other forms of data movement like query result sets also need to be considered. The questions raised while creating the temporary table are as follows −

- Where is that temporary table to be held?
- How do you make such table visible?

We should avoid the accidental flouting of security restrictions. If a user with access to the restricted data can generate accessible temporary tables, data can be visible to non-authorized users. We can overcome this problem by having a separate temporary area for users with access to restricted data.

Documentation

The audit and security requirements need to be properly documented. This will be treated as a part of justification. This document can contain all the information gathered from −

- Data classification
- User classification
- Network requirements
- Data movement and storage requirements
- All auditable actions

Impact of Security on Design

Security affects the application code and the development timescales. Security affects the following area −

- Application development
- Database design
- Testing

Application Development

Security affects the overall application development and it also affects the design of the important components of the data warehouse such as load manager, warehouse manager, and query manager. The load manager may require checking code to filter record and place them in different locations. More transformation rules may also be required to hide certain data. Also there may be requirements of extra metadata to handle any extra objects.

To create and maintain extra views, the warehouse manager may require extra codes to enforce security. Extra checks may have to be coded into the data warehouse to prevent it from being fooled into moving data into a location where it should not be available. The query manager requires the changes to handle any access restrictions. The query manager will need to be aware of all extra views and aggregations.

Database design

The database layout is also affected because when security measures are implemented, there is an increase in the number of views and tables. Adding security increases the size of the database and hence increases the complexity of the database design and management. It will also add complexity to the backup management and recovery plan.

Testing

Testing the data warehouse is a complex and lengthy process. Adding security to the data warehouse also affects the testing time complexity. It affects the testing in the following two ways −

- It will increase the time required for integration and system testing.
- There is added functionality to be tested which will increase the size of the testing suite.

BACKUP & RECOVERY

A data warehouse is a complex system and it contains a huge volume of data. Therefore it is important to back up all the data so that it becomes available for recovery in future as per requirement. In this chapter, we will discuss the issues in designing the backup strategy.

Backup Terminologies

Before proceeding further, you should know some of the backup terminologies discussed below.

- **Complete backup** − It backs up the entire database at the same time. This backup includes all the database files, control files, and journal files.
- **Partial backup** − As the name suggests, it does not create a complete backup of the database. Partial backup is very useful in large databases because they allow a strategy whereby various parts of the database are backed up in a round-robin fashion on a day-to-day basis, so that the whole database is backed up effectively once a week.
- **Cold backup** − Cold backup is taken while the database is completely shut down. In multi-instance environment, all the instances should be shut down.
- **Hot backup** − Hot backup is taken when the database engine is up and running. The requirements of hot backup varies from RDBMS to RDBMS.
- **Online backup** − It is quite similar to hot backup.

Hardware Backup

It is important to decide which hardware to use for the backup. The speed of processing the backup and restore depends on the hardware being used, how the hardware is connected, bandwidth of the network, backup software, and the speed of server's I/O system. Here we will discuss some of the hardware choices that are available and their pros and cons. These choices are as follows −

- Tape Technology
- Disk Backups

Tape Technology

The tape choice can be categorized as follows −

- Tape media
- Standalone tape drives
- Tape stackers
- Tape silos

Tape Media

There exists several varieties of tape media. Some tape media standards are listed in the table below −

Tape Media	Capacity	I/O rates
DLT	40 GB	3 MB/s
3490e	1.6 GB	3 MB/s
8 mm	14 GB	1 MB/s

Other factors that need to be considered are as follows −

- Reliability of the tape medium
- Cost of tape medium per unit
- Scalability
- Cost of upgrades to tape system
- Cost of tape medium per unit
- Shelf life of tape medium

Standalone Tape Drives

The tape drives can be connected in the following ways −

- Direct to the server
- As network available devices
- Remotely to other machine

There could be issues in connecting the tape drives to a data warehouse.

- Consider the server is a 48node MPP machine. We do not know the node to connect the tape drive and we do not know how to spread them over the server nodes to get the optimal performance with least disruption of the server and least internal I/O latency.

- Connecting the tape drive as a network available device requires the network to be up to the job of the huge data transfer rates. Make sure that sufficient bandwidth is available during the time you require it.
- Connecting the tape drives remotely also require high bandwidth.

Tape Stackers

The method of loading multiple tapes into a single tape drive is known as tape stackers. The stacker dismounts the current tape when it has finished with it and loads the next tape, hence only one tape is available at a time to be accessed. The price and the capabilities may vary, but the common ability is that they can perform unattended backups.

Tape Silos

Tape silos provide large store capacities. Tape silos can store and manage thousands of tapes. They can integrate multiple tape drives. They have the software and hardware to label and store the tapes they store. It is very common for the silo to be connected remotely over a network or a dedicated link. We should ensure that the bandwidth of the connection is up to the job.

Disk Backups

Methods of disk backups are −

- Disk-to-disk backups
- Mirror breaking

These methods are used in the OLTP system. These methods minimize the database downtime and maximize the availability.

Disk-to-Disk Backups

Here backup is taken on the disk rather on the tape. Disk-to-disk backups are done for the following reasons −

- Speed of initial backups
- Speed of restore

Backing up the data from disk to disk is much faster than to the tape. However it is the intermediate step of backup. Later the data is backed up on the tape. The other advantage of disk-to-disk backups is that it gives you an online copy of the latest backup.

Mirror Breaking

The idea is to have disks mirrored for resilience during the working day. When backup is required, one of the mirror sets can be broken out. This technique is a variant of disk-to-disk backups.

Note – The database may need to be shutdown to guarantee consistency of the backup.

Optical Jukeboxes

Optical jukeboxes allow the data to be stored near line. This technique allows a large number of optical disks to be managed in the same way as a tape stacker or a tape silo. The drawback of this technique is that it has slow write speed than disks. But the optical media provides long-life and reliability that makes them a good choice of medium for archiving.

Software Backups

There are software tools available that help in the backup process. These software tools come as a package. These tools not only take backup, they can effectively manage and control the backup strategies. There are many software packages available in the market. Some of them are listed in the following table –

Package Name	**Vendor**
Networker	Legato
ADSM	IBM
Epoch	Epoch Systems
Omniback II	HP
Alexandria	Sequent

Criteria for Choosing Software Packages

The criteria for choosing the best software package are listed below –

- How scalable is the product as tape drives are added?
- Does the package have client-server option, or must it run on the database server itself?
- Will it work in cluster and MPP environments?
- What degree of parallelism is required?
- What platforms are supported by the package?
- Does the package support easy access to information about tape contents?
- Is the package database aware?
- What tape drive and tape media are supported by the package?

SERVICE LEVEL AGREEMENT (SLA)

A service level agreement (SLA) is essential to the design process of the data warehouse. In particular, it is essential to the design of the backup strategy. You need the SLA as a guide to the rates of backup that are required, and more importantly the rate at which a restore needs to be accomplished. The SLA affects not just the backup, but also such fundamental design decisions as partitioning of the fact data.

Definition of Types of System

There are a number of terms that are frequently used when talking about large systems such as data warehouses:

- Operational
- Mission critical
- 7 x 24
- 7 x 24 x 52

The meaning of these terms should be clear from the names, but the terms are often loosely used — some would say even misused — so we shall define what we mean by them.

- An operational system is a system that has responsibilities to the operations of the business.
- A mission critical system is a system that the business absolutely depends on to function.

- A 7 x 24 system is a system that needs to be available all day every day, except for small periods of planned downtime.
- A 7 x 24 x 52 system is true 7 x 24 systems, which is required to be running all the time.

Defining the SLA

These topics can be further divided into two categories :

- User requirements
- System requirements

User requirements are the elements that directly affect the users, such as hours of access and response times. System requirements are the needs imposed on the system by the business such as system availability.

User Requirements

One of the key sets of requirements to capture during the analysis phase is the user requirements. Detailed information is required on the following:

- User online access — hours of work
- User batch access
- User expected response times
- Average response times
- Maximum acceptable response times

System Requirements

The key system requirement that needs to be measured is the maximum acceptable downtime for the system. These questions can be asked from the viewpoints both of user discomfort and of business damage. However, the reality is that in systems of this size and complexity you have to work to the business damage limit, not the user discomfort limit, otherwise costs complexity soar. The SLA needs to stipulate fully any measures related to downtime. In particular, some measure of the required availability of the server is needed.

Availability can be measured in a number of different ways, but however you state it in the SLA it must be unambiguous. It is normal for availability to be measured as a required percentage of uptime. In other words:

$$D = 100 - A$$

where D = acceptable downtime, and A is the percentage required availability.

Note 'that D covers both planned and unplanned downtime. This can be further broken down into acceptable online downtime (Du), and acceptable offline downtime (D1). These can be defined as.

$$D_{„} - 100 — A_n$$

$$D_1 = 100 — A_1$$

Where an if the percentage of N for which the system is required to be available; A is the percentage of 24-N) hours for which the system is required to be available; and N is the number of hours in the user/online day.

<u>OPERATING THE DATA WAREHOUSE</u>

The operation of a data warehouse is a non-trivial task. The design of the operational environment is no less challenging., the design is not made any easier by the need for operations to be as automated as possible.

Day-To Day Operations of the Data Warehouse

The daily operations of the data warehouse are equally as comp but less fraught than the overnight processing. The main purpose 1 daytime usage of the machine is to service the users' queries Some other operations that can occur during the day are:

• Monitoring application daemons • Query management • Batch • Ad hoc • Backup/recovery • Housekeeping scripts • Data feeds • Drip feed of load files • Data extracts • Query results • Data marts • Performance monitoring • Event management • Job scheduling

The warehouse application managers are likely to have a number of service or background process running. These processes are often called daemons. These daemons are used to log events, act as servers and so on.

Query management is a vital part of the daily operations. Some of the monitoring and control can be automated, but there will still be a requirement for a DBA to be available to deal with any problems.

Backups are usually avoided during the user day, to avoid contentions with the users, queries. In addition, in a data warehouse the bulk of the data is read-only, and there is little that happens during working hours that will require backing up.

There are always housekeeping tasks the need to be run. Scripts that purge down directories, log off idle connections and so on are as much a part of a data warehouse as of any other system.

The data feeds may arrive at any time of the day. Having some of the data arrive during the day is not an issue unless it causes network problems.

Data extracts are subsets of data that are offloaded form the server machine onto other machines. Extracts can be unscheduled user extracts off some query results, or they can be scheduled extracts such as data mart refreshes. .

Extraction of user query results can impose a significant load on the network, and has to be controlled.

Data marts are regular and, as such, more controllable data extracts. as with other extracts, they will incur network costs, unless they are transferred via some medium such as tape.

Performance monitoring, event monitoring and scheduling are all ongoing and highly automate operations. They should require minimal manual input to maintain. Any manual interfacing should be via a menu system or the relevant GUI front end.

Logins for each user will need to be created, maintained, and deleted. This is a task that should be menu driven to prevent accidents. Adding a user with the wrong privileges can prove to be costly mistake. The users should all fall into easy categories or groups, allowing template login to be created for each group. This template should have all the default accesses, profiles, roles and so on for that group.

Log files, which log ongoing events, and trace files, which dump information on specific processes, will generate a vast amount of data, and can occupy a log of disk space. Some trace files will have errors or race information from unsolved problems; other may have performance statistics form some recent test runs.

Some trace information is worth keeping in its own right

Starting up and shutting down of the server, database or the data warehouse applications are tasks that are likely to he performed infrequently. They are, however, tasks that it is important to get right, because shutting down the machine or database incorrectly can cause problems on restart. Each will need scripted and menu-driven procedures for shutting it down, and starting it up.

Printing requirements will need to be established. From experience, printing is not a common data warehouse requirement. Sets of most require are too large to printing meaningfully.

Problem management is an area that needs to be clearly defined and documented, it is vital that all administration staff know who, where and when to call if problems arise. It is common to have several groups inside an organization responsible for different parts of the data warehouse.

Upgrading a data warehouse is always a problem. As the system becomes a victim of its own success, it will become harder and harder to find extended periods of time when the system can be down.

Overnight Processing

There are a number of major issues that need to be addressed, if this is not to become a stumbling block to the success of the data warehouse.

The key issue is keeping to the time window and ensuring that you do not eat into the next business day. The sheer volume of work that has to be accomplished, and the serial nature of many of the operation, makes this more difficult than it may first seem.

The tasks that need to be accomplished overnight are, in order:

1. Data rollup, 2. Obtaining the data 3. Data transformation 4. Daily load. 5. Data cleanup 6. Index creation 7. Aggregation creation and—maintenance, 8. Backup 9. Data Archiving 10. Data mart refresh.

In data rollup order data is rolled up into aggregated from to save space. This operation is not dependent on any of the other operations in the list.

The first real step in the overnight processing is obtaining the data. This in itself is probably a simple operation; the problems arise when the data is delayed or cannot be transferred.

Data transformation this will vary from data warehouse to data warehouse and it is possible that you will have no transformation at all to perform.

The load step is again a potential serialization point in the overnight processing. The load itself can be parallelized, but the data cleanup cannot process until the data is in the database.

The data cleanup processing required will again vary from data warehouse to data warehouse.

Aggregation creation and maintenance, cannot, really process until all the data has arrives, and has been transformed, loaded and cleaned.

The final overnight operation in most systems will be the backup, ['his has to be last, because some of the most important items that need be backup are the new and modified aggregations. The backup can by interlaced with other operations, because the newly loaded data can his backup as soon as it hits the database.

If data has to be archived, this process will usually get run as part or the backup. Depending on the frequency of the data archiving, this on prove to be a major overhead.

Data marts and other data extracts- if they exist- may have to he refreshed on a regular basis. You need to ascertain exactly when suvii extracts need to be available, if possible, and how regularly they need I be refreshed.

UNIT – V Planning , Tuning and Testing

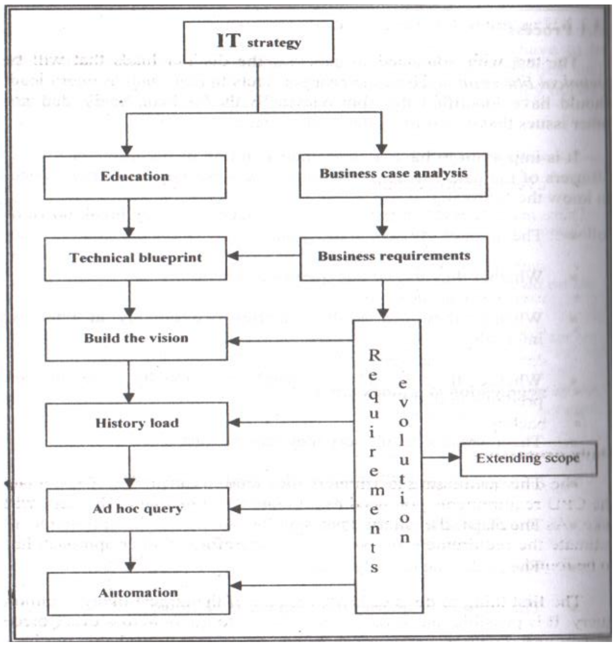
CAPACITY PLANNING

Process

The capacity plan for a data warehouse is defined within the technical blueprint stage of the process. The business requirements stage should have identified the approximate sizes for data, users, and any other issues that constrain system performance.

It is important to have a clear understanding of the usage profiles of all users of the data warehouse for each user or group of users you need to know the following:

- The number of users in the group;
- Whether they use ad hoc queries frequently
- Whether they use ad hoc queries occasionally at unknown intervals
- Whether they use ad hoc queries occasionally at regular and predictable times
- The average size, of query they tend to run;
- The maximum size of query they tend to run;
- The elapsed login time per day;
- The peak time of daily usage;
- The number of queries they run per peak hour;
- The number of queries they run per day.



Estimating the Load

A When choosing the hardware for the data warehouse there arc intItiy things-Ito- consider such as hardware architecture, resilience. and so on, One poilif to6-feriiember is that the data warehouse will probably groW rapidly from its initial configuration, so it is not sufficient to consider 1110 initial size of the data warehouse.

Initial Configuration

When sizing the initial configuration you will have. no information or statistics to work with, and the sizing will need to be done on the predicted load. Estimation this load is difficult, because there is an as of element to it. However, if the phased approach is taken, the as hoc element will be grown to meet the demand.

How much CPU bandwidth?

To start with you need to consider the distinct loads that will be placed on the system. There are many aspects to this, such as query load, data load, backup and so on, but essentially the load can be divided into two distinct phase: • daily processing • overnight processing There are discussed in part Three, but essentially they break down as follows;

• daily processing • user query processing • overnight processing • data transformation and load • aggregation and index creation • backup

Daily processing

The daily processing is centered around the user queries. To estimate the CPU requirements you need to estimate the time that each query will take. As much of the query load will be ad hoc it is impossible to estimate the requirement of every query; therefore another approach has to be found.

The first thing to do is estimate the size of the largest likely common query. It is possible that some user will want to query across every piece of data in the data in the data warehouse, but this will probably not be a common requirement. It is more likely that the users will want to query the most recent week or month's worth of data.

The progress any further we need to know the I/O characteristics of the devices that the data will reside on. This allows us to calculate the scan rate S at which the fact data can be read. This will depend on the disk speeds and on the throughput ratings of the controllers. Clearly this also depends on the size of f itself.

Using S and F you can calculate T, the time in seconds to perform a full table scan of the period in question.,

$$T = F/S$$

In fact you should calculate a number of times, Ti-ta, which depend on the degree of parallelism that you are using to perform the scan. Therefore we get

$$T1 = F/ Si$$

$$Tn= F /Sn$$

where s 1 is the scan speed of a single disk or striped disk set, and Sn

is the scan speed of all the disks or disk sets that F is spread across. **Overnight processing**

The CPU bandwidth required for the data transformation will depend on the amount of data processing that is involved. Unless there is an enormous amount of data transformation, it is unlikely that this operation will require more CPU bandwidth than the aggregation and index creation operation.

If users are allowed to leave queries running overnight, or if queries are allowed to run for 24 hours or more, you will also need to make a separate allowance for that processing, above and beyond the calculated CPU requirement for the overnight processing.

How Much Memory?

Memory is commodity that you can never have enough of. What you need to estimate is the minimum requirements. There are a number of things that affect the amount of memory required.

First, there are the database requirements. The database will need memory to cache data blocks as they are used; it will also need memory 41 to cache parsed SQL statements and so on. There requirements will vary from RDBMS to RDBMS, and you will need to work them out for whatever software you are using.

Secondly, each user connected to the system will use an amount of memory: how much will depend on how they are connected to the system and what software they are running. As it is likely that the users will be connected in a client-server mode the user memory requirement may be quite small

Finally, the operating system will require an amount of memory. This will vary with the operating system and the features and tools you are running. You can get the hardware vendor to estimate how much memory the system will use.

How much disk?

The largest calculation you will need to perform is the amount of disk space required. This can be a very tricky thing to calculate, and how successfully you do it will depend on how successfully the analysis captures the requirements.

The disk requirement can be broken down into the following categories:

- database requirements
- administration
- fact and dimension data
- aggregations
- non — data requirements
- operating system requirements
- other software requirements
- data warehouse requirements
- User requirements.

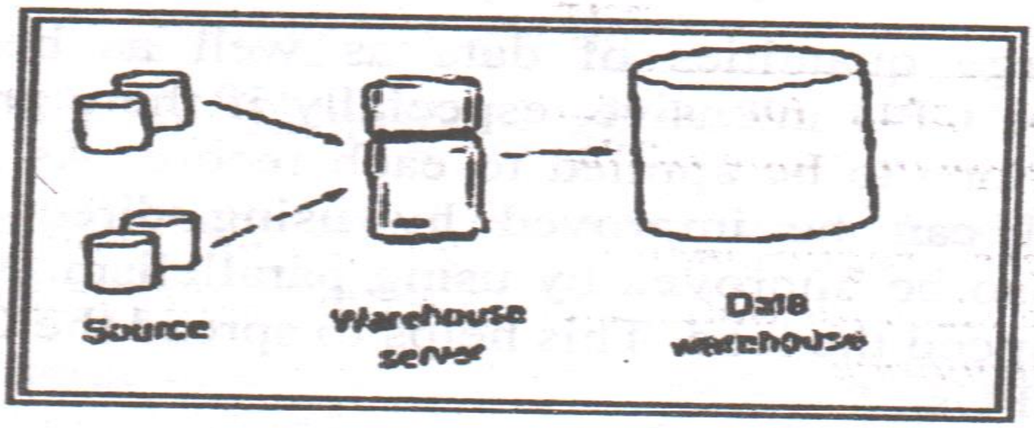
Database sizing

The database will occupy most of the required disk space, and when sizing it you need to be sure you get it right. There are a number of aspects to the database sizing that need to be considered.

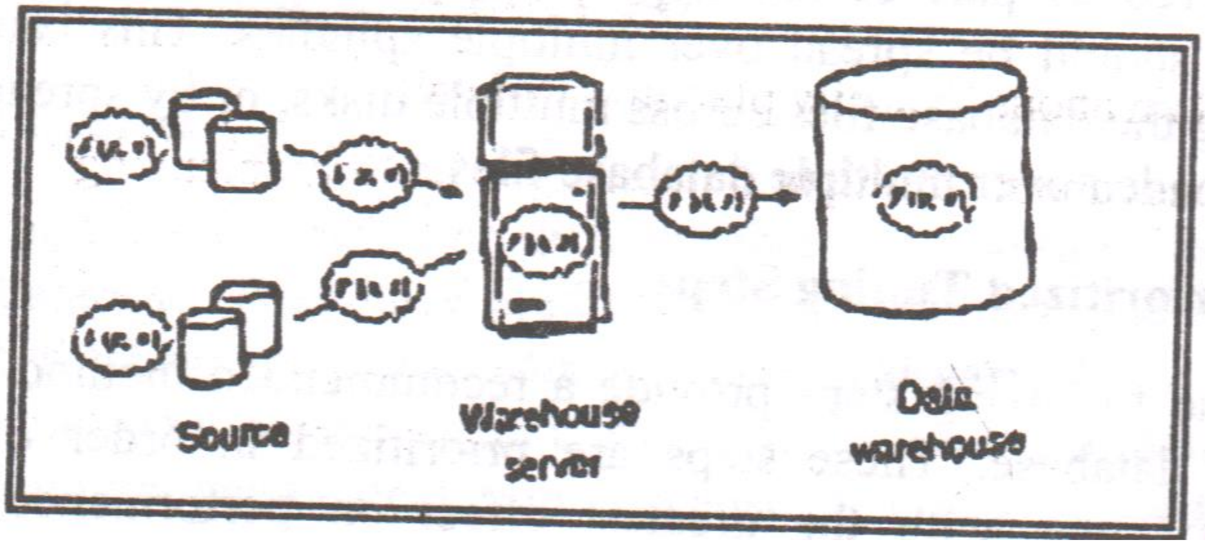
Tuning the Data warehouse

Tuning the Data Load

Because the data load is the entry point into the system, it provides le first opportunity to improve performance. Essentially the flow of data as depicted in figure.



As shown in figure, there are opportunities to transform the data at very stage of this data flow. If all data transformation or integrity checks occur before the data arrives at the data warehouse system, all you need to know is the exception time of arrival. If the checks are performed on the data warehouse system, either before after the data is loaded. or indeed a combination of both, they will have a direct effect on the capacity and performance of the system. Any tuning of the integrity checks will depend on what form they take.



It is not unknown to apply rudimentary checks to the data being loaded. For example, if the data is telephone call records. You may want to check the each call has a valid customer identifier. If the data is retail information, you may want to check whether the commodity being sold has a valid product identifier. One point to note is that just because some of the loading data fails such a check it does not necessarily mean that the data is invalid for the call records, it could easily be that the data is correct and that the customer information is not up to data. This can happen when a new customer is added as a service subscriber. They will probably use the phone immediately the service is connected, but it may take several days for the customer information to trickle through to the data warehouse from the customer systems.

There is little that can be done to tune any business rules enforced by constrains. If the rules are enforced by using SQL or by trigger code that code needs to be tuned to maximal efficiently.

Loading large quantities of data as well as being a heavy I/O operation can be CPU intensive, especially if there are a lot of checks and transformations to be applied to each record. As mentioned above, the load speed can be improved by using direct load techniques. The load can also be improved by using parallelism. Multiple processes can be used to speed the load. This helps to spread the CPU load amongst any available CPUs.

If using multiple loads, care is needed to avoid introducing I/O bottlenecks. Ensure that the source data file does not become a source of contention. Ideally, the source data should be split into multiple files so that each load process can have its own source file. The source files should also be spread over multiple disks to avoid contention on a single disk or I/O controller.

The load destination can also become a bottleneck, particularly if a large number of load processes are being used. This needs to be considered as part of the original database design. Each table to be loaded should be spread over multiple spindles. This can be done by striping the database files across multiple disks, or by spreading the table to be loaded over multiple database files.

Prioritized Tuning Steps

The following steps provide a recommended method for tuning an oracle database. These steps are prioritized in order of diminishing returns: steps with the greatest effect on performance appear first.

For optimal results, therefore, resolve tuning issues in the order listed: from the design and development phase through instance tuning.

Step 1: Tune the business rules
Step 2: Tune application design
Step 3: Tune the data design
Step 4: Tune the logical structure of the database
Step 5: Tune database operations
Step 6: Tune the access paths
Step 7: Tune memory allocation
Step 8: Tune I/O and physical structure
Step 9: Tune resource contention
Step 10: Tune the underlying platforms

After completing these steps, reassess your database performance and decide whether further tuning is necessary.

Tuning is an iterative process. Performance gains made in later steps may pave the way for further improvements in earlier steps, so additional passes through the tuning process may useful.

Tuning Queries

The data warehouse will contain two types of query.

1. Fixed queries
2. Ad hoc queries

Both types of query need to be tuned, but they require different techniques dealing with them.

Fixed queries

Tuning the fixed queries is no different than the traditional tuning of a relational database. They have predictable requirements, and it can be tested to find the best execution plan. The only real variables with these queries are the size of the data being queried and the data skew. Even these variables can be dealt with, because the query can be tested as often as desired.

AD HOC queries

The number of users of the data warehouse will have a profit effect on the performance, in particular if they are ad hoc tog It is important have a clear understanding of the usage profiles of all re users. For each user or group of users you need to know the following

Data Warehousing and Data mining

- The number of users in the group
- Whether they use ad hoc queries frequently
- Whether they use ad hoc occasionally at unknown intervals
- Whether they use ad hoc queries occasionally at regular predictable times.

- The average size of query they tend to run.
- The maximum size of query they tend to run
- The elapsed login time per day
- The peak time of daily usage
- The number of queries they run per peak hour;
- Whether they require drill-down access to the base data.

TESTING THE DATA WAREHOUSE

As with any development project, a data warehouse will need, only more so the complexity and size of data warehouse system comprehensive testing both more difficult and more nt The fact that queries that take minutes to run on small scale testing sufficient. It is necessary to establish that queries scale with the upshot is that comprehensive testing of a data warehouse take time. This makes predicting test schedules difficult.

The Testing Terminologies

The various terminologies during are:

1. Unit testing: to ensure that the component or module and behaves as per the specified requirements.

2. System testing: to ensure that the interactions between the components or module function correctly as per the specified requirements.

3. Integration testing : to test the solution from an end-to-end perspective and ensure that the system works in a production-like environment

4. Acceptance testing; to verify that the entire solution meets the business requirements and successfully supports the business processes from a user's perspective.

5. System assurance testing; to ensure and verify the operational readiness of the system in a production environment. This is also referred to as the warranty period coverage.

Beside the above mentioned testing phases, there are a few other popular categories of testing phases.

White box testing: knowledge of program structure and business rules is used formulate test cases,. Also known as glass box, structural, clear box and open box testing. A testing technique whereby explicit knowledge of the internal working of the item (program or software) is being tested is used to select the test data.

Black box testing: appropriate for system testing, where the system is considered as a black box. The test classes are derived from the requirements. This technique is also known as functional testing where in the internal working of the item (program and software) are being tested us bit by the tester.

Testing the operational environment

Testing of the data warehouse operational environment is another key set of tests that will have to be performed. There are a number of aspects that need to be tested:

- Security
- Disk configuration
- Scheduler
- Management tools
- Database management

Testing is very important for data warehouse systems to make them work correctly and efficiently. There are three basic levels of testing performed on a data warehouse −

- Unit testing
- Integration testing
- System testing

Unit Testing

- In unit testing, each component is separately tested.
- Each module, i.e., procedure, program, SQL Script, Unix shell is tested.
- This test is performed by the developer.

Integration Testing

- In integration testing, the various modules of the application are brought together and then tested against the number of inputs.

- It is performed to test whether the various components do well after integration.

System Testing

- In system testing, the whole data warehouse application is tested together.
- The purpose of system testing is to check whether the entire system works correctly together or not.
- System testing is performed by the testing team.
- Since the size of the whole data warehouse is very large, it is usually possible to perform minimal system testing before the test plan can be enacted.

Test Schedule

First of all, the test schedule is created in the process of developing the test plan. In this schedule, we predict the estimated time required for the testing of the entire data warehouse system.

There are different methodologies available to create a test schedule, but none of them are perfect because the data warehouse is very complex and large. Also the data warehouse system is evolving in nature. One may face the following issues while creating a test schedule −

- A simple problem may have a large size of query that can take a day or more to complete, i.e., the query does not complete in a desired time scale.
- There may be hardware failures such as losing a disk or human errors such as accidentally deleting a table or overwriting a large table.

Note − Due to the above-mentioned difficulties, it is recommended to always double the amount of time you would normally allow for testing.

Testing Backup Recovery

Testing the backup recovery strategy is extremely important. Here is the list of scenarios for which this testing is needed −

- Media failure
- Loss or damage of table space or data file
- Loss or damage of redo log file
- Loss or damage of control file
- Instance failure
- Loss or damage of archive file
- Loss or damage of table

- Failure during data failure

Testing Operational Environment

There are a number of aspects that need to be tested. These aspects are listed below.

- **Security** – A separate security document is required for security testing. This document contains a list of disallowed operations and devising tests for each.
- **Scheduler** – Scheduling software is required to control the daily operations of a data warehouse. It needs to be tested during system testing. The scheduling software requires an interface with the data warehouse, which will need the scheduler to control overnight processing and the management of aggregations.
- **Disk Configuration.** – Disk configuration also needs to be tested to identify I/O bottlenecks. The test should be performed with multiple times with different settings.
- **Management Tools.** – It is required to test all the management tools during system testing. Here is the list of tools that need to be tested.
    - Event manager
    - System manager
    - Database manager
    - Configuration manager
    - Backup recovery manager

Testing the Database

The database is tested in the following three ways –

- **Testing the database manager and monitoring tools** – To test the database manager and the monitoring tools, they should be used in the creation, running, and management of test database.
- **Testing database features** – Here is the list of features that we have to test –
    - Querying in parallel
    - Create index in parallel
    - Data load in parallel

- **Testing database performance** – Query execution plays a very important role in data warehouse performance measures. There are sets of fixed queries that need to be run regularly and they should be tested. To test ad hoc queries, one should go through the user requirement document and understand the business completely. Take time to test the most awkward queries that the business is likely to ask against different index and aggregation strategies.

Testing the Application

- All the managers should be integrated correctly and work in order to ensure that the end-to-end load, index, aggregate and queries work as per the expectations.
- Each function of each manager should work correctly
- It is also necessary to test the application over a period of time.
- Week end and month-end tasks should also be tested.

Logistic of the Test

The aim of system test is to test all of the following areas –

- Scheduling software
- Day-to-day operational procedures
- Backup recovery strategy
- Management and scheduling tools
- Overnight processing
- Query performance

Note – The most important point is to test the scalability. Failure to do so will leave us a system design that does not work when the system grows.

DATA WAREHOUSE FUTURES

Following are the future aspects of data warehousing.

- As we have seen that the size of the open database has grown approximately double its magnitude in the last few years, it shows the significant value that it contains.
- As the size of the databases grow, the estimates of what constitutes a very large database continues to grow.
- The hardware and software that are available today do not allow to keep a large amount of data online. For example, a Telco call record requires 10TB of data to be kept online, which is just a size of one month’s record. If it requires to keep records of sales, marketing customer, employees, etc., then the size will be more than 100 TB.
- The record contains textual information and some multimedia data. Multimedia data cannot be easily manipulated as text data. Searching the multimedia data is not an easy task, whereas textual information can be retrieved by the relational software available today.
- Apart from size planning, it is complex to build and run data warehouse systems that are ever increasing in size. As the number of users increases, the size of the data warehouse also increases. These users will also require to access the system.
- With the growth of the Internet, there is a requirement of users to access data online.

Hence the future shape of data warehouse will be very different from what is being created today.