3rd class

Advanced Databases

مدرس المادة: م.م نور حيدر عبد الامير

اعداد: د.عماد كاظم
م.م نور حيدر

Review

1. What is file systems
2. Disadvantage of file systems
3. Requirement to computer system
5 . What is Data Base ?
6 . What is DBMS?
7 . What is Data models ?
8 . What is advantage of Data Base ?
9 . Data Base Architecture?

1. File System:

The file system is typically described as various files and numbers of different application programs are written to extract records from and add records to the appropriate files

.

2. Traditional files features and limitation:

1- When we handle a lot of files in one application program **access becomes complicated and various kind of trouble often Occur**.

2- Since data to description correspond to programs one to one it is **complication to update all program when data format is update**.

3- Since one data is often duplicated in more than two files when is updated **it becomes inconsistent unless maintained simultaneously to all file concerned.**

4- One file is usually created in a suitable format for some times **the file format cannot be used for another application**.

5- One file may be **restricted to only one access key**.

6- **Programmer must consider file integrity** to avoid HW and SW troubles.

3. File System Disadvantage:-

1- Data redundancy and inconsistency existed.
2- Difficulty in accessing data.
3- Data isolation.
4- Concurrent access anomalies.
5- Security problems existed.
6- Integrity problems existed.

4. Requirement to Computer System

1- Since users application are new more complicated, we **want to operating system to manage the relations among several files**.

2- The amount of data increased so much , hens the management of files is now very complex. we **want to reduce the numbers of file and make them clear and neat.**

3- We **want to consider one data item from various view point**.

4- We **want to decrease the manpower for program development and maintenance.**

5- The end user **wants access (retrieve-use) the necessary data quickly and by himself.**

5. Database:

It is a collection of interrelated data store together without harmful or unnecessary redundancy data to serve multiple application. The data is stored so as to be independent from the programs.

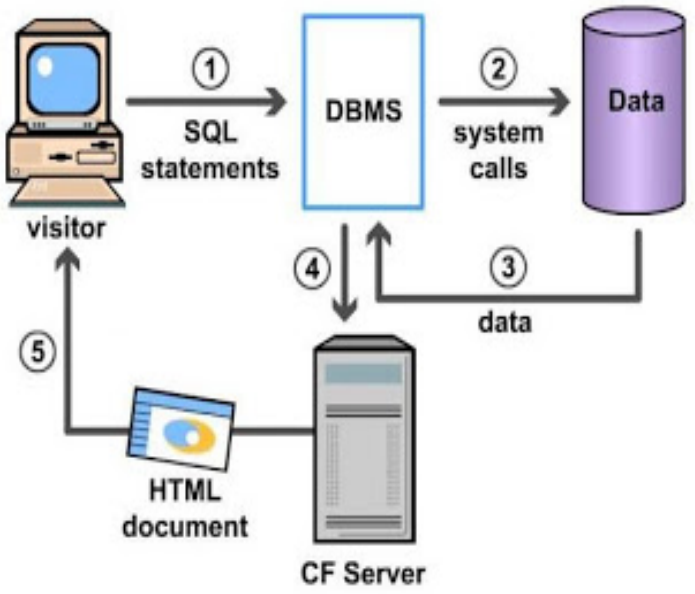
6. Database Advantages

1- Reduction in data redundancy.
2- The ability to operate on deferent data structure.
3- Independent of data from the program.
4- High speed of retrieval and fast on line.
5- High degree of flexibility in handling data format.
6- Minimum cost.
7- Inconsistent can be avoided.
8- Integrity can be maintained.
9- Standard parameter can be enforced.
10- Security restriction can be applied.

Why high speed of retrieval data in database?

- Data base using direct access.
- using Index keys.
- Data linking together one with other.
- Redundancy is existed in some time.

Database Management System (DBMS)



8. DBMS:

It is a software package designed to store and manage database to gets:

- Data independence and efficient access.
- Reduced application development time.
- Data integrity and security.
- Uniform data administration.
- Concurrent access and recovery

9. Schema:

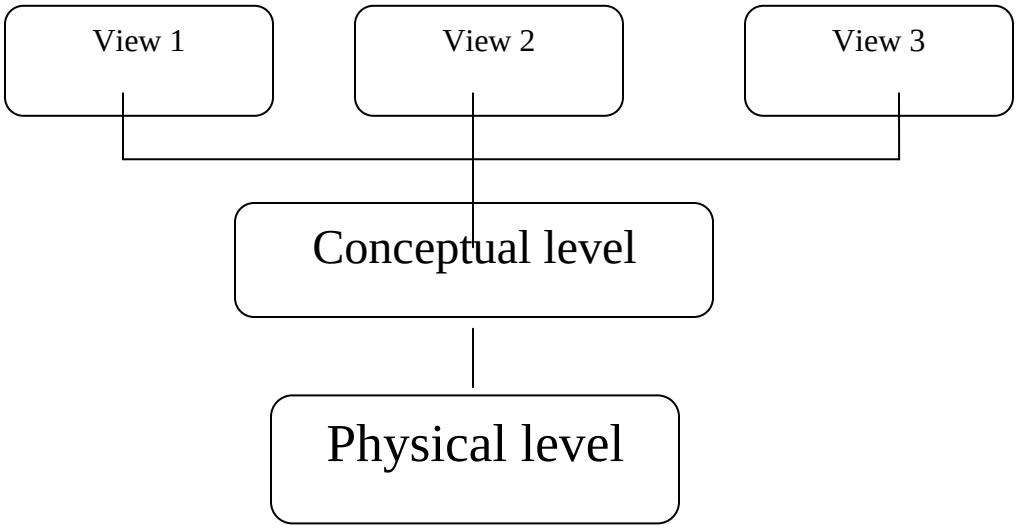
Is a description of a particular collection of data, using the given data model.

10. Data Abstraction

DBMS:Is a collection of interrelated files and set of programs that allow to access and modify these files.

the major purpose of DB system is to provide users with an abstract view of the data ,that is the system hides certain details of how the data is stored and maintained .however, in order for the system to be usable ,data must be retrieved efficiently . this concern has led to the design of complex data structure for the representation of data in the DB since many DBS users are not computer trained ,the complexity is hidden from them through several levels abstraction in order to simplify their interaction with the system .

- **Physical level:**
  The lowest level of abstraction describes ***how*** the data are actually stored in details
- **Conceptual level:**
  the next higher level of abstraction describes ***what*** data are actually stored in
  the data base and the relationships that exit among the data
- **View level**
  The highest level of abstraction describes only part of the entire database.
  The system may provide many views for the same database



The three levels of data abstraction

Example:
Let the following records with no and name for each one

1- Customer

Cust-no : number
Cust-name : string

2- Account
Acc-no : integer
Acc-name : char

3- Employee
emp-no : number
emp-name : string

At the physical level:
Each record: Customer, Account and Employee can described as block of consecutive storage (for example byte,words,…)

At the conceptual level
Each record: Customer, Account and Employee is described by a type definition, illustrated above and the interrelationship among these record type

At the View level
Several view of the data base are defined

11. Data Models

Is a collection of conceptual tools for describing data, data relationships, data semantic, and consistency constraints. The various data models that have been proposed fall into three different groups:

1 Object-based logical models.
2 Record-based logical models.
3 Physical data models.

- Object based logical models are used in describing data at the conceptual and view level. They are characterized by the fact that they provide fairly flexible structuring capabilities and allow data constraints to be specified explicitly Such as:

14. Relational model:

Relational model is important model which is represents data relationships among data by a collection of tables each of them has number of columns and rows with unique table names. Columns represent the fields or the attributes in the table and rows represents the records or entities in the table.

Employee No.	Employee Name	Sex Code	Certification Code	Department No.	Tel No.

Certification Code	Certification name

Department No	Department Name

15. Distributed database Overview

In recent years the availability of database and of computer networks has given rise to a new field- distributed database. A distributed database is, in brief, an integrated database which is built on top of a computer network rather than on a single computer . In a distributed database system, the database is stored on several computer, the computers communicate with one another through various communication media, such as high-speed buses or telephone lines. They do not share main memory, nor do they share a clock. The processor in distributed system may vary in size and function. They may include small microcomputers, workstations, minicomputers, and large general purpose computer systems. These

processors are referred to by a number of different names, such as sites, nodes, and computers, depending on the context in which they are mentioned. We mainly use the term site, in order to emphasize the physical distribution of these systems.

A distributed database system consists of a collection of sites, each of which may participate in the execution of transaction which access data at one site, or several sites. The main difference between centralized and distributed database systems is that, in the former, the data resides in one single location, while in the latter, the data resides in several locations. This distribution of data is the cause of many difficulties that will be addressed .

16. Why distributed data base?

There are many reasons imply to DDB such as:

1- Organizational and E-economic reasons.

2- Interconnection of existing data base.

3- Capacity and incremental growth.

4- Reliability and availability.

5- Local autonomy.

6- Efficiency and flexibility.

7- Data sharing and distributed control

8- Improved Performance

17. Terminology and Concept

1- Data Base / data base instance
These terms are often used interchangeably but there are not the same things

- Database: is the set of physical files containing data. These files comprise table spaces, redo logs, and control files.

- Database instance: is the set of processes and memory structure that manipulate a DB .

- Note: DB may be accessed by one or more DB instance.
- Note: DB instance may access exactly one DB.

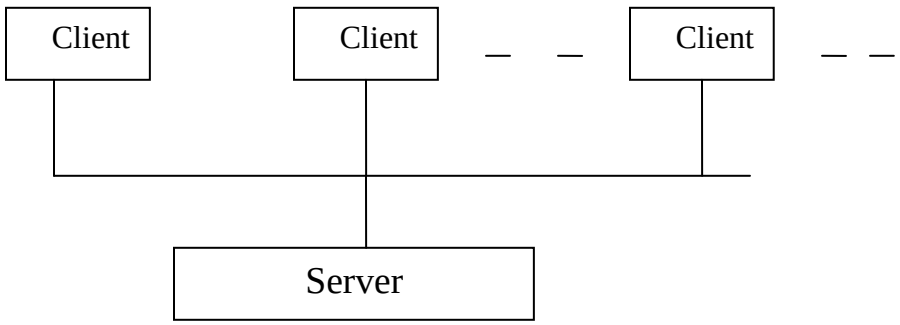
2- Centralized Database system:
It is a DB system that run are single computer system and do not interact with other computer system.

3- Single user system:
Is a desktop, unit used by a single person usually with only one CPU and one or two hare desks and usually only one person using the machine at a time.

4- Multi user system:
Is a system that has more disks and more memory, may have multiple CPUs and has multi-user OS. DB systems designed for use single users usually do not provide many of the facilities that multi-user DB provides. In particular they may not support concurrency control which is not required when only single user can generate up date.

5- Client – system:
It is typically PCs computer running web browsers, the client role is to provide an interface to the user (such as developer 2000 in oracle) and web browsers. Or the client is the application or soft wares that initiates the connection it may be an end user application such as web page, or it may be another oracle server.

6. Server System:
The server system is the system which is satisfies requests generated by client system. Can be categorized as:



1- transaction – server (query –server) provide an interface to which clients can send requests to perform an action , in response to which they execute the action and send back results to the client .

2- Data –server :
Allow client to interact with the servers by making request to read or update data, in units such as files or pages. For example files servers provide a file system. Interface where clients can create, update, read, and delete files. Data server for DB systems offer much more functionality, they support units of data, such as page, tupelos, or objects that are smaller than file. They provide indexing facilities for data and provide transaction facilities so that the data are never left in an inconsistent state if client machine or process fails.

18. Functionality Database system

Functionality provided by DB system can be divided in to two parts:

1- The back end which is manages access structures, query evolution and optimization, concurrency control and recovery.

2- The front end of DB consist of tools such as : SQL user interface, form interface, report generation tools and data mining and analysis tools.

The interface between front-end and back-end is through SQL or through an application program.

19. Structure of Distributed database

The definition of distributed database (DDB) include two important aspects:

1. Distribution :
The fact that the data are not resident at the same (processor) so that we can distinguish a distributed database from a single centralized database.

2. Logical correlation :

the fact that the data have some properties which use them together, so that we can distinguish a distributed database from a set of local

database or files which are resident at different sites of a computer network.

The problem with above definition is that both properties, distributions and logical correlation, are too vaguely defined to always discriminate between those cases.

3. Local application :

   The database must be local database as well as its new properties of distribution .

4. Global application :

   Any database to distributed database must have at least one global database

In order to develop a more specific definition , let us consider a few examples:

Example 1
Consider a bank that has three branches at different locations. At each branch, a computer controls the teller terminals of the branch and the account database at one branch. Computer are connected by a communication network (see figure-1).

Example 2
Consider the same bank of previous example, with a system configuration as show in figure-2 . The same processors with their databases have been moved away from the branches to a common building and are now connected with a high-band width local network.

Example 3
Consider the same bank of the previous example but with the system configuration show in figure-3 . The data of the different branches are distributed on three "backend" computers, which perform the database management functions. The application programs are executed by a different computer, which requests database access services from back ends when necessary.

20. Distributed database system:

It is a collection of distributed local database system which have logical correlation , local application ,global application and distributed on multi site.

21. A distributed database system consists

A distributed database system consists of a collection of sites, each of which maintains a local database system . Each site is able to process local transactions, those transactions that access data only in the single site. In addition, a site may participate in the execution of global transaction, those transactions that access data in several sites. The execution of global transaction requires communication among the sites. The sites in the system can be connected physically in a variety of ways. The various topologies are represented as graphs whose nodes correspond to sites. An edge from node A to node B corresponds to a direct connection between the two sites. Some of the most common configurations are depicted in figure-4 .

The major differences among these configurations involve:

1. Installation cost, the cost of physically linking the sites in the system.

2. Communication cost, the cost in time and money to send a message from site A to site B.

3. Reliability, the frequency with which a link or site fails.

4. Availability, the degree to which data can be accessed despite the failure of some links or sites.

As we shall see, these differences play an important role in choosing the appropriate
mechanism for handling the distribution of data .

The sites of a distributed database system may be distributed physically either over a large geographical area(such as the United States) or over a small

geographical area (such as a single building or as a number of adjacent buildings). The former type of network is referred to us a long-haul network, the latter is referred to us a long-area network .

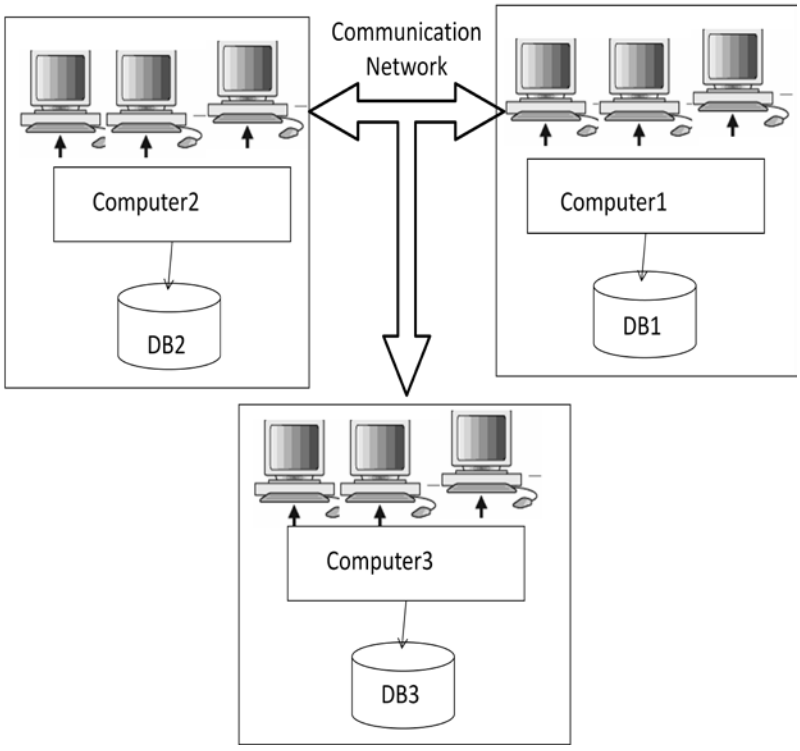


Fig (1) Distributed Database for global application

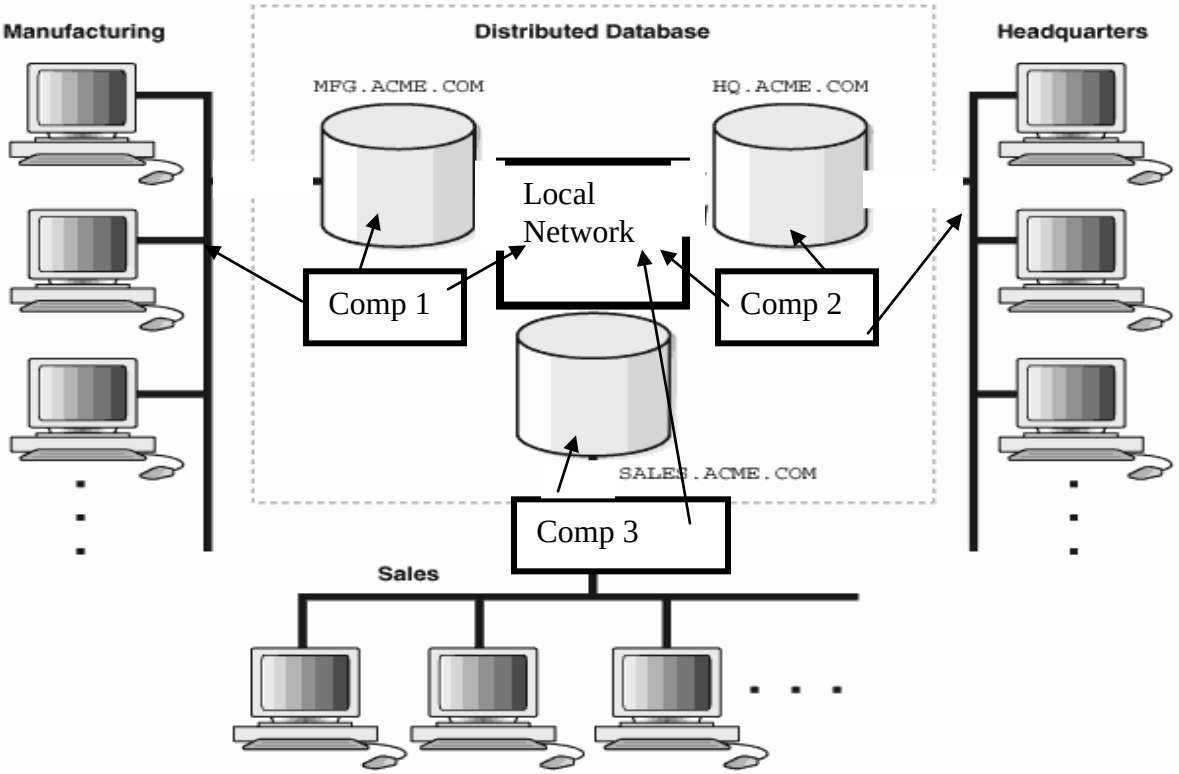


Fig (2) Distributed Database on Local Network LAN

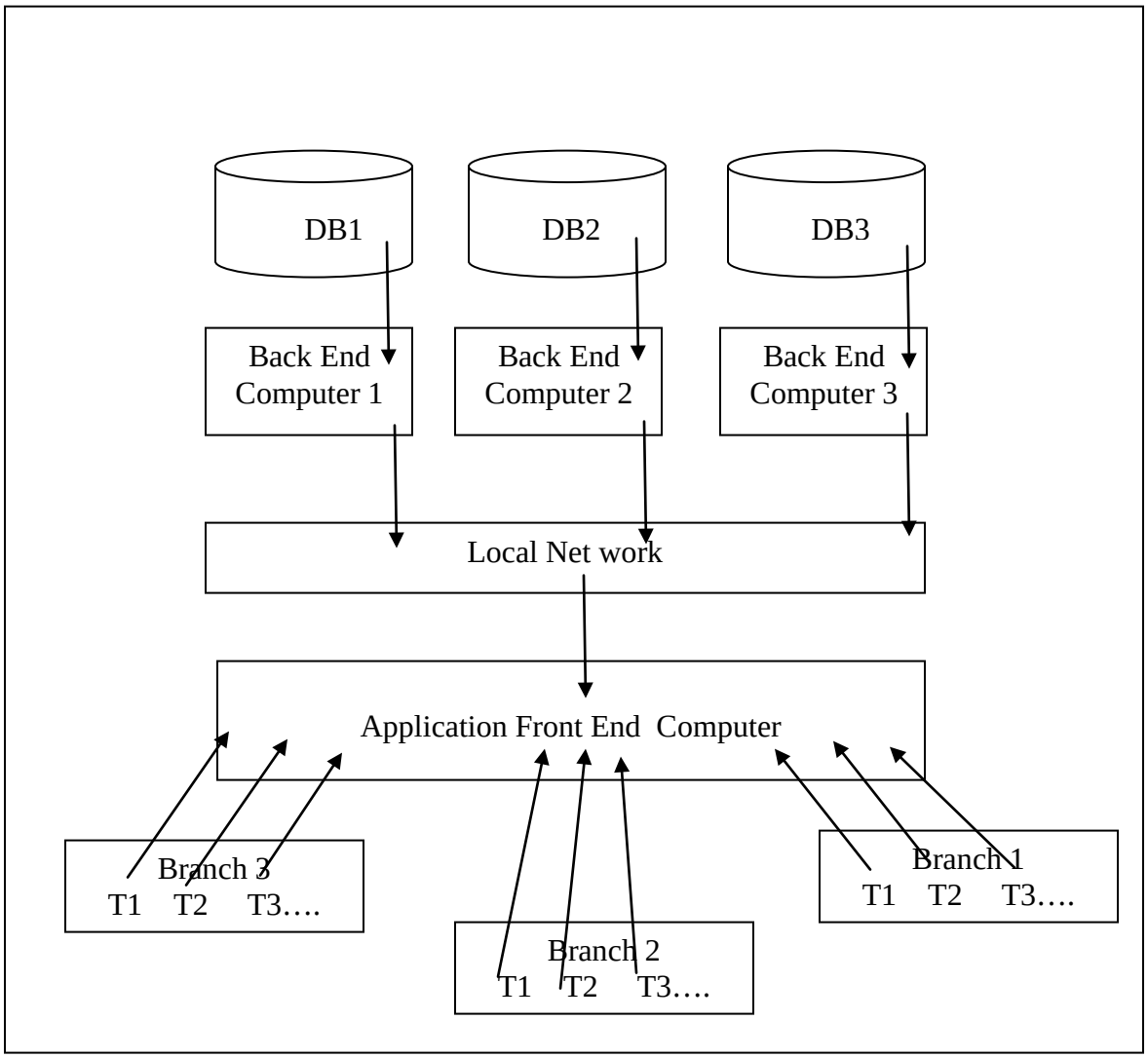
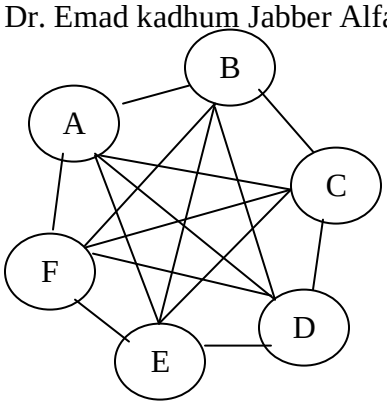
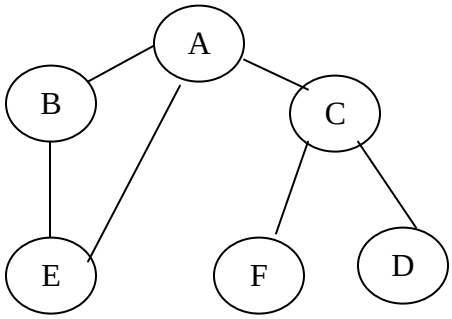


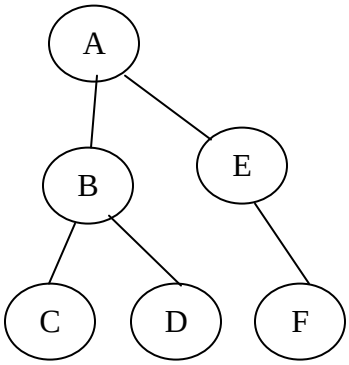
Fig (3) Multi processor system



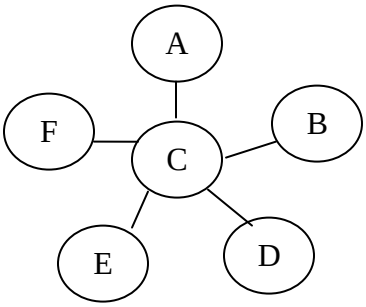
Fully Connection Network



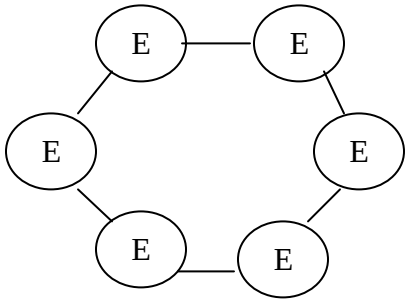
Partially Connection Network



Tree Structured Network



Star Network



Ring Network

Fig (4) Network Topology

22. Features of distributed versus centralized database

A Distributed databases are present different features from traditional (centralized system) so that it is useful to look at the typical features of traditional database and compare them with the features of distributed database. The features which characterize the traditional database approach are:

1. Centralized control

The possibility of providing centralized control over the information resources of whole enterprise or organization was considered of the strongest motivations for introducing database, they were developed as the evolution of information systems in which each application had its own private files. The fundamental of a database administrator(DBA) was to guarantee the safety of data itself was recognized to be an important investment of the enterprise which required a centralized responsibility. In distributed database, the idea of centralized control is much less emphasized. In general, in DDB it is possible to identify a hierarchical control structure based on a global database administrator, who has the central responsibility of the whole database and on local database administrators, who have the responsibility of their respective local database.

2. Data independence

Means that the actual organization of data is transparent to the application program, having conceptual view of data called conceptual schema. These programs are unaffected by changes in the physical organization of data. In distributed database , data independence has the same importance as in centralized database , however , a new aspect is added to the usual notion of data independence , namely, distribution transparency . By distribution transparency we mean that programs can be written as if the database were not distributed. Thus the correctness of programs is unaffected by the movement of data from one site to another, however , their speed of execution is affected.

3. Reduction of redundancy

In traditional database, redundancy was reduced as far as possible for two reasons:

first, inconsistencies among several copies of the same logical data are automatically avoided by having only one copy, and second, storage space is

saved by eliminating redundancy. Reduction of redundancy was obtained by data sharing , by allowing several applications to access the same files and records. In distributed database, however, there are several reasons for considering data redundancy as a desirable feature: first, the locality of applications can be increased if the data is replicated at all sites where applications need it, and second, the availability of the system can be increased, because a site failure does not stop execution of applications at other sites if the data is replicated.

4. Complex physical structures and efficient access

The reason for providing complex accessing structures is to obtain efficient access to the data. In distributed database, complex accessing structures are not the right tool for efficient access because cannot be provided by using physical structures and it is very difficult to build and maintain such structures.

5. Integrity, recovery, and concurrency control

The database integrity is the means of atomic transactions, it is a sequence of operations which either are performed completely or are not performed at all. In DDB transaction atomicity has a particular flavor: when the system have two sites, the first site is operational and the second site is not operational and when we must transfer application from first site to the second , the transaction should

be aborted. The atomicity property is ensured by the various recovery and concurrency control schemes . When we are dealing with a distributed database system , since several sites may be participating in its execution . The failure of one of these sites, or the failure of a communication link connecting these sites, may result in erroneous computations.

6. Privacy and security

In traditional database, the database administrator, having centralized control, can ensure that only authorized access to the data is performed . But in DDB , local administrators are faced with the same problem as database administrator in traditional database.

23 Advantages of database distribution

The primary advantage of DDB systems is the ability to share and access data in a reliable and efficient manner. The following is a list of the main advantages:

1. Data Sharing and distributed control If a number of different sites are connected , then a user at one site may be able to access data available at another site. For example, in the distributed banking system it is possible for a user in one branch to access data in another branch. Without this capability, a user wishing to transfer funds from one branch to another would have to resort to some external mechanism that would, in effect, be a single centralized database. The primary advantage of sharing data by means of data distribution is that each site is able to retain a degree of control over data stored locally. In a centralized system, the database administrator of the central site controls the database. In a distributed system, there is a global database administrator responsible for the entire system. A part of these responsibilities is delegated the local database administrator for each site. Depending upon the design of the distributed database system, each administrator may have a different degree of local autonomy. The possibility of local autonomy is often a major advantage of distributed database.

2. Reliability and Availability

If one site fails in a distributed system, the remaining sites may be able to continue operating . If data item replicated in several sites, a transaction needing a particular data item may find it in any of several sites. Thus, the failure of a site does not necessarily imply the shutdown of the system.

The failure of one site must be detected by the system, and appropriate action may be needed to recover from the failure. Finally, when the failed site recovers or is repaired, mechanisms must be available to integrate it smoothly back into the system.

3. Speed of Query processing

If a query involves data at several sites, it may be possible to split the query into subqueries that can be executed in parallel. In distributed system , there is no sharing of main memory, so not all of the join strategies for parallel processors can be applied directly to distributed systems. In those cases in which data is replicated queries may be directed by the system to the least heavily loaded sites.

4. Incremental growth

If an organization grows by adding new, relatively autonomous organizational units then the distributed database approach supports a smooth incremental growth with a minimum degree of impact on the already existing units.

5. Reduced communication overhead

In a geographically distributed database, the fact that many applications are local reduce the communication overhead with respect to a centralized database. Therefore, the maximization of the locality of applications is one of primary objectives in distributed database design.

6. Performance considerations

The existence of several processors results in the increase of performance through a high degree of parallelism.

24 Disadvantages of Database distribution

The primary disadvantage of distributed database systems is the added complexity among the sites. This increased complexity takes the form of:

1. software development cost It is more difficult to implement a distributed database system and, thus, more costly.
2. Greater potential for bugs

   Since the sites that comprise the distributed system operate in parallel, it is harder to ensure the correctness of algorithms. The potential exist for extremely subtle bugs .
3. Increased processing overhead.

   The exchange of messages and the additional computation required to achieve

   interstice coordination are a form of overhead that does not arise in centralized systems.

25 Fundamental principle of distributed DB

the user in distributed system should be able to behave exactly as if the system were not distributed because distributed system have:

1- Local autonomy.
2- No reliance on a central site.
3- Continuous operation.
4- Location independence.
5- Fragmentation independence.

6- Replication independence.
7- Distributed query processing.
8- Distributed transaction management.
9- Hardware independence.
10- Network independence.
11- DBMS independence.

1- Local autonomy :-

The sites in distributed system should be autonomous since Local autonomy means that all operations at a given site are controlled by that site. No site depends on other site for its successful operation.

Local autonomy also implies that local data is locally owned and managed with local accountability. All data belongs to some local database even if it is accessible from other sites. Such matters as integrity, security, and physical storage representation of local data these remain under the control of the local site.

2- No reliance on a center site:-

local autonomy implies that all sites must be treated as equals ,so there must not be any reliance on central site. Reliance on a central site would be undesirable for at least the following reasons:-

1- Central site may be a bottleneck.
2- The system would be vulnerable if the central site went down.
3- Continuous operation.

These point can provide greater reliability and greater availability.

- Reliability: - is the probability that the system is up and running at any given moment. Reliability is improved in distributed system because such systems can continue to operate in the face of failure of some individual component such as individual site.

- Availability: - is the probability that the system is up and running continuously throughout specified period reliability and availability is improved in distributed system because of possibility of data replication.

4- Location independence.

It also known as location transparency the idea of transparency is: The users should not have to know where data is physically stored. But rather should be able

to behave at least from logical standpoint as if the data were all stored at their own local site. Location independence is desirable because it simplifies application program and end user activities, such as it allow data to migrate from site to site. And that allows data to be moved around the network in response to changing performance requirements.

5- Fragmentation independence.

The data can divided into pieces or fragments for physical storage purpose and distinct fragment can be stored at different sites. Fragment is important for performance reasons. Data can be stored at the location where it most frequently used. So that most operation are local and network traffic is reduced. Look to the following example:

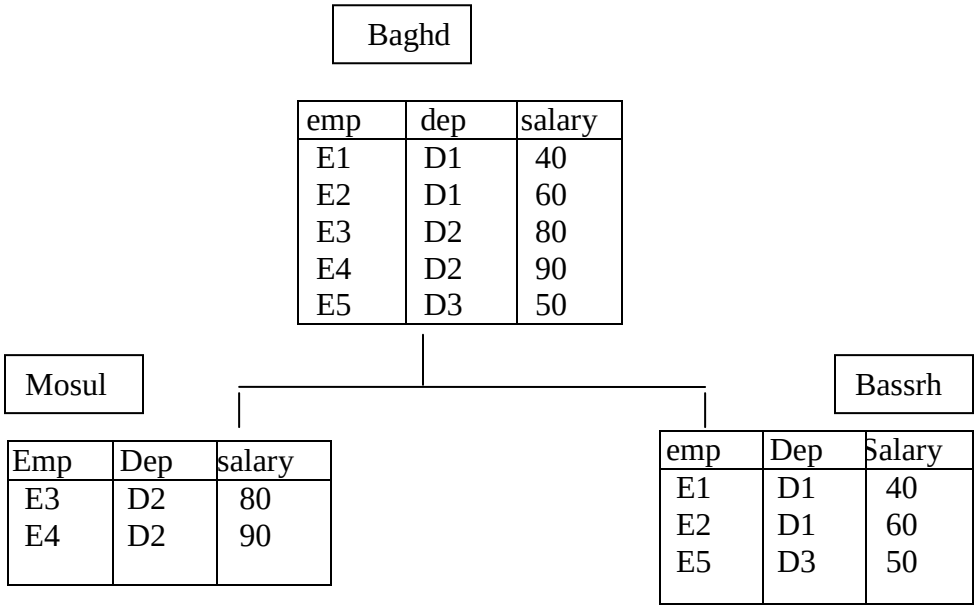


Fig ( ) Example of fragmentation (User perception)

There are basically three kinds of fragmentation:

- Horizontal fragmentation.
- Vertical fragmentation.
- Mixed fragmentation.

Fragmentation independence also known as fragmentation transparency and it like location transparency is desirable because it simplifies application programs

and end user activities, such as it allows the data to be defragmentation at any Time.

Fragmentation transparency implies that users will be presented with a view of the data in which the fragments are logically recombined by means of suitable joins and unions. It is the responsibility of optimizer to determine which fragments need to be physically access in order to satisfy any given user request.

6- Replication independence

It means a given fragment can be represented in storage by many distinct copies or replicas. Stored at many distinct sites. Replication is desirable because :

1- it can mean better performance application can operate on local copies instead of having to communication with remote sites.

2- It can also mean better availability the processing at lease for retrieval as long as at least one copy remains available.

The disadvantage of replication ,is that when a given replicated object is update ,all copies of that object must be update which call *update propagation*.

7- Distributed query processing

There are two points to be made under this heading:

- First , consider the query “get London suppliers of red parts.” Suppose the user is at the new York site and the data is at the London site .suppose too that there are **n** suppliers that satisfy the request. If the system is relational , the query will basically involve two messages: one to send the request from new York to London, and one to return the result set of n tuples from London to new York , if on the other hand , the system is not relational but a record -at-a-time system , the query will basically involve 2n messages: n from new York to London requesting “the next” supplier, and n from London to new York to return that “next” supplier. The example thus illustrates the point that a relational distributed system is likely to outperform a non relational one by possibly order of magnitude.
- Second, optimization is even more important in a distributed system than it is in a centralized one , the basic point is that in a query such as the one mentioned in the previous paragraph that involves several sites, there will be

many possible ways of moving data around the system in order to satisfy the request , and it is crucially important that an efficient strategy be found . for instance, a request for(say) the union of a relation rx stored at site X and a relation ry stored at site Y could be carried out by moving rx toY or by moving ry to X or by moving both to a third site Z(etc). A compelling illustration of this pint. involving the same query once again ("get system supplier numbers for London suppliers of red parts"),is presented one from distributed database.

8- Distributed transaction management

There are two major aspects to transaction management, recovery and concurrency , and both require extended treatment in the distributed environment . in order to explain that extended treatment, it is first necessary to introduce a new term , agent . In a distributed system, a single transaction can involve the execution of code at many sites; in particular , it can involve updates at many sites. Each transaction is therefore said to consist of several, Agents**,** where an agent is the process performed on behalf of a given transaction at a given site. And the system clearly needs to know when two agents are part of the same transaction; for example, two agents that are part of the same transaction must obviously not be allowed to deadlock with each other. Turning now to recovery specifically in order to ensure that given transaction is atomic (all or nothing) in the distributed environment, the system must clearly ensure that the set of agents for that transaction either all commit in unison or all roll back in unison. This effect can be achieved by means of the two-phase commit protocol, already discussed, through not in the distributed context, we will have more to say regarding two-phase commit for a distributed system . As for concurrency : concurrency control in must distributed system is typically based on locking . just as it is in non distributed system.(some products use multi-version control instead [16.1],but conventional locking still seems to be the technique of choice in most system.) we will discuss this topic also in a little more detail .

9- Hardware independence

All, real-world computer installation typically involve a multiplicity of different machines like IBM machines, fujitsu machines, HP machines, PCs and workstations of various kinds, and so on . and there is a real need to be able to integrate the data on all of those systems and present the user with a "single – system image". Thus , it is desirable to be able to run the same DBMS on

different hardware platforms, and furthermore to have those different machines all participate as equal partners in a distributed system.

10- Operating system independence

This objective is partly a corollary of the previous one, and also does not really require much discussion here. it is obviously desirable, not only to be able to run the same DBMS on different hardware platforms, but also to be able to run it on different operating system platforms as well including different operating systems on the same hardware and have an OS/390 version and a UNIX version and a Windows version all participate in the same distributed system.

11-Network independence

Once again there is not much to say ; if the system is to be able to support many disparate sites, with disparate hardware and disparate operating system , it is obviously desirable to be able to support a variety of disparate communication networks also.

12-DBMS independence

Under this heading, we consider what is involved in relaxing the strict homogeneity assumption. That assumption is arguably a little too strong : All that is really needed is that the DBMS instances at different sites all support the same interface __ they do not necessarily all have to be copies of same DBMS software .for example ; if Ingres and Oracle both supported the SQL standard, then it might be possible to get an Ingres site and an Oracle site to talk to each other in the context of a distributed system. in other words , it might be possible for the distributed system to be heterogeneous, at least to some degree. Support for heterogeneity is definitely desirable . the fact is , real-world computer installations typically run not only many different machines and many different operating systems, they very often run different DBMSs as well ;and it would be nice if those different DBMSs culd all participate somehow in a distributed system in other words , the ideal distributed system should provide DBMS independence. This is such a large topic ,however ,and such an important one in practice , that we devote a separate section to it.

26 Trade-off in distributing the database

There are several reasons for building distributed data base system such as:

1- Sharing of data.
2- Reliability and availability.

3- Speed of query processing

However along with this advantage come several disadvantage such as:

1- Software development cost.
2- Greater potential for bugs.
3- Increased processing overload.

27 Distributed data base management system DDBMS

DDBMS: It is a software package supports to creation and maintenance of distributed database were developed by the centralized data base management system the software components which are typically necessary for building a distributed database are:

1- The database management component DBM.
2- The data communication component DC.
3- The data dictionary DD. Which is extended to represent information about the distributed of data in network.
4- The distributed data base component DDB.

so the DDBMS is the set of above four components the services which supported by the DDBMS are:

1- Remote data base access.
2- Some degree of distributed transparency.
3- Support data base administration DA and control this feature include tools for monitoring the data, gathering information data base utility and providing global view of data and file existing at the viruses sites.
4- Support concurrency and recovery for distributed transaction.

Type of DDBMS:

There are two type of DDBMS

- **Homogeneous DDBMS**: it mean that the DDBMS with the same DBMS at each site even if the computers (and/or) the operating system are not the same.
- **Heterogeneous DDBMS**: it mean the DDBMS with at least two different DBMS so it cause the problem of translating between the different data models of different local DBMS to the complexity of homogenous DDBMS.

Other definition

- DDB: a set of database in distributed system that can appear to application as a single data source.
- distributed processing: the operation that occurs when an application distributed its tasks among different computers in a network.

EXERCISES:

1- What we mean by distributed data base?
2- Explain with examples the structure of DDB?
3- What is the various topologies configuration use for physically connected sites in DDB? And what are the differences between then?
4- What are the differences between long-haul network and local-area network?
5- What are the differences between local transaction and global transaction?
6- What are the main differences between DDB and centralized DB?
7- What we mean by reduction of redundancy integrity, recovery, and concurrency control?
8- What are the main advantages of DDB?
9- What are the main disadvantages of DDB?
10- What we mean by DDBMS?
11- What is the software components used for building DDB?
12- What are the services which are supported by software components in DDB?
13- What we mean by homogenous DDB? And what are the differences with heterogeneous DDB?
14- What we mean by : DDB ,distributed processing ?

28 Distributed Database Architecture

Architecture of DDB consist of different levels of distribution transparency, these levels are conceptually relevant in order to understand distributed database , and allows us to determine easily the different between levels of distribution transparency. Figure (6) shows a reference architecture for a distributed database.

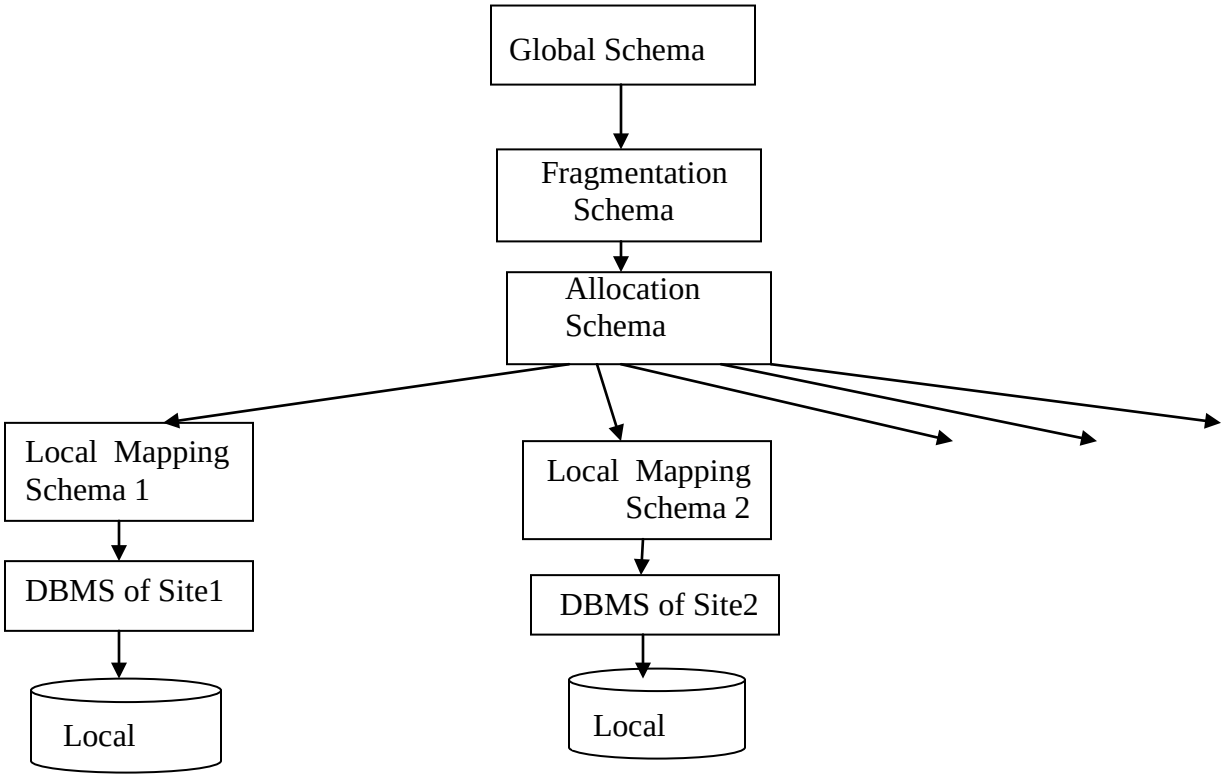


Figure (6) A reference architecture for distributed database

This reference architecture is not explicitly implemented in all distributed database, however, its levels are conceptually relevant in order to understand the organization of any distributed database.

At the top level of figure (6) is the global schema.

-The global schema defines all the data which are contained in the DDB as if the database were not distributed at all.

-We will use the relational model for the definition a global schema. Using this model, the global schema consists of the definition of a set of global relations.

-Each global relation can be split into several non overlapping portions which are called fragments.

-There are several different ways in which to perform the split operation, they are described in the next section.

-Fragments are logical portions of global relations which are physically located at one or several sites of the network.
-The allocation schema defines at which site(s) a fragment is located.

This architecture provides a very general conceptual framework for understand distributed database. ***The three important objectives which motivate the features of this architecture are the*** :

1. Separating the concept of data fragmentation from the concepts of data allocation. This separation allows us to distinguish two different levels of distribution transparency, namely
   -fragmentation transparency and
   -location transparency.
   The system minimize degree to which a user needs to be aware of how a relation is stored.
   A system can hide the details of the distribution of data in the network and we call this network transparency.
   Fragmentation transparency is the highest degree of transparency and consists of the fact that the user or application programmer works on global relations .
   Location transparency is a lower degree of transparency and required the user or application programmer to work on the fragments instead of global relations, however he does not know where the fragments are located.

2. Explicit control of redundancy. The reference architecture provides explicit control of redundancy at the fragment level. As we shall see, the explicit control over redundancy is useful in several aspects of distributed database management.

3. Independence from local DBMSs .This feature called local mapping transparency , allows us to, study several problems of DDBMS without having to take into account then specific data models of local DBMSs. In a homogeneous system it is possible that the site independent schema are defined using the same data model as local DBMSs , thus reducing the complexity of this mapping. Another type of transparency which is related with location transparency is replication transparency. Replication transparency means the user unaware of the replication of fragments. Replication transparency is implied by location transparency and it is possible that the user has no location transparency but has replication transparency.

The decomposition of global relation into fragment can be performed by applying two different types of fragmentation : Horizontal fragmentation and Vertical fragmentation . We will first considered these two types of fragmentation separately and then consider the more complex fragmentation which can be obtained by applying composition of both Mixed fragmentation.

29 Fragmentation

Fragmentation mean: the Data can be divided into pieces or fragments for physical storage purposes and distinct fragments can be stored at different sits. Fragmentation are desirable for performance reasons :data can be stored at the location where it is most frequently used, so that most operations are local and network traffic is reduced. The decomposition of global relation into fragment can be performed by applying difference types of fragmentation such as Horizontal fragmentation , Vertical fragmentation and mixed fragmentation.

30 Horizontal fragmentation

consists of partitioning the tuples of a global relation r into subsets r1,r2,….,rn each subset can contain data with common properties. The reconstruction of relation r can be obtained by taking the union of all fragments, that is : r = r1 U r2 U …… U rn

For example, suppose that the relation r is the deposit relation of figure (7)this relation has only two branches, Baghdad and Mosul , and if

we choose the attribute branch-name for horizontal fragmentation the relation, then the result are two different fragment show in figure (8).

branch-name	account-number	customer-name	balance
Baghdad	305	Salem	500
Baghdad	226	Ahmad	336
Mosul	177	Ahmad	205
Mosul	402	Hassan	1000
Baghdad	155	Hassan	62
Mosul	408	Hassan	1123
Mosul	639	Ali	750

Figure (7)Sample deposit relation

branch-name account-number customer-name balance

Baghdad	305	Salem	500
Baghdad	226	Ahmad	336
Baghdad	155	Hassan	62

---

deposit1

---

branch-name account-number customer-name balance

Mosul	117	Ahmad	205
Mosul	402	Hassan	1000
Mosul	408	Hassan	1123
Mosul	639	Ali	750

---

deposit2

Figure (8) Horizontal fragmentation of relation deposit by using branch-name

31 Vertical Fragmentation

Vertical Fragmentation for global relation is the subdivision of its attributes into groups, subdivision is accomplished by adding a special attribute called a tuple-id to the scheme R. A tuple-id is a physical or logical address for a tuple Since each tuple in r must have a unique address, the tuple-id attribute is a key for the scheme.

Deposit-scheme3 = (branch-name, customer-name, tuple-id)

Deposit-scheme4 = (account-number, balance, tuple-id)

-

branch-name account-number customer-name balance tuple-id

--

Baghdad	305	Salem	500	1
Baghdad	226	Ahmad	336	2
Mosul	177	Ahmad	205	3
Mosul	402	Hassan	1000	4

Baghdad	155	Hassan	62	5
Mosul	408	Hassan	1123	6
Mosul	639	Ali	750	7

--

Figure (9) The deposit relation of figure (7)with tuple-id

---

branch-name	customer-name	tuple-id
Baghdad	Salem	1
Baghdqd	Ahmad	2
Mosul	Ahmad	3
Mosul	Hassan	4
Baghdad	Hassan	5
Mosul	Hassan	6
Mosul	Ali	7

---

deposit3

Account-number	balance	tuple-id
305	500	1
226	336	2
177	205	3
402	1000	4
155	62	5
408	1123	6
639	750	7

Deposit4

Figure (10) Vertical fragmentation of relation deposit

32 Mixed Fragmentation

The relation r is divided into n number of fragment relations r1,r2,…..,rn. Each fragment is obtained as the result of applying either the horizontal fragmentation or vertical fragmentation scheme on relation r or a fragment of r which was obtained previously.

To illustrate, suppose that the relation r is the deposit relation of Figure 2.2 This relation is divided initially into the fragments deposit3 and deposit4 as defined above. We can further divide fragment deposit3 by using horizontal fragmentation scheme into deposit3a and deposit3b.Thus relation r is divided into three fragments deposit3a , deposit3b , and deposit4 each of these may reside in a different site.

33 Framework For Distributed Database Design

Consider a relation **r** that is to be stored in data base .there are several issues Involved in storing this relation in distributes database including.

- Replication

The system maintains several identical replicas(copies)of the relation each replica is stored in a different site, resulting in data replication.

- Fragmentation: the relation is partitioned in to several fragments. Each fragment is stored in a different site.

- Replication and fragmentation:

This is combination of the above .two so the system maintains several identical replicas of each such fragment.

In distributed database there are four problems connected with process design. Two of them become

1. ***The design of the global schema and***
2. ***The design of local physical database at each site***.

The techniques which can applied to these problems are the same as in centralized database . The distribution of the database adds to the above problems two new once;

3. ***Designing the fragmentation and***
4. ***Designing the allocation of fragments.***

The distinction between these two problem is conceptually relevant, since the first one deals with the logical aspect which motivate the fragmentation of a global relation, while the second one deal with the physical placement of data at the various sites.

34 Objectives of design of data distribution

In the design of data distribution , the following objectives should be taken into account:

processing locality

distributing data to maximize processing

locality corresponds to the simple principle of placing data as possible to the applications which use them.

The simplest way of characterizing processing locality is to consider two types of references to data: "local" references and "remote" references. Availability and reliability of distributed data A high degree of availability and reliability by storing multiple copies of the same information, the system must be able to swish to an alternative copy when the one that should be accessed under normal conditions is not available. Reliability is also achieved by storing multiple copies of the same information since it is possible to recover from crashes or from the physical destruction of one of the copies by using the other copies.

Workload distribution , distributing the workload over the sites is an important feature of distributed computer systems. Workload distribution is done in order to maximize the degree of parallelism of execution of applications. Storage cost and availability database distribution should reflect the cost and the availability of storage at the different sites . The cost of data storage is not relevant if compare with CPU, I /O and transmission cost of applications, but the limitation of available storage At each site must be considered. Using all the above criteria at the same time is extremely difficult, since this leads to complex optimization models. It is possible to consider some of the above features as constraints, rather than objective .

In the following, we will use the simple approach of maximizing processing locality this objective is adequate to indicate design criteria which are general and of practical use.

35 Top-down and Bottom-up approaches to design of data distribution

There are two alternative approaches to the design of data distribution,

1. the top-down and
2. the bottom-up approaches.

In the Top-down approach, we start by designing the global schema, and we proceed by designing the fragmentation of database, and then by allocating the fragments to the sites, creating physical images. The approach is completed by performing, at each site, the physical design of the data which are allocated to it. This approach is the most attractive for systems which are developed from start point, since it allows performing the design logically. When the distribution database is developed as the clustering of databases, it is not easy to follow the top-down approach. In this case the global schema is often produced as a compromise between the existing data descriptions. When existing database are aggregated or clustered.

a Bottom-up approach to the design of data distribution can be used . This approach is based on the integration of existing schemas into single global schema. By integration we means the merging of common data definitions and resolution of conflicts among different representations given to the same data. When existing database are clustered into a distributed database it is possible that they use different DBMSs . As we have discussed in chapter 1 a heterogeneous system adds to complexity of data integration the need for a translation between different representation . In summary, the bottom-up design of distributed database requires:

1. The selection of a common database model for describing the global schema of the database.
2. The translation of each local schema into the common data model.
3. The integration of the local schemas into a common global schema

36 The allocation of fragments

The data allocation problem has been widely analyzed in several references and use several models which build the optimal allocation of files with different objectives There are some criteria that can be used for allocating fragments, these criteria are:

37 General criteria for fragment allocation

In determining the allocation of fragment, it is important to distinguish whether we design a final non redundant or redundant allocation. Determining a non redundant final allocation is easier. The simplest method is a " best-fit " approach, a measure is associated with each possible allocation, and the site with the best measure is selected.

Replication introduces further complexity in the design, because:

1. The degree of replication of each fragment becomes a variable of the problem.
2. Modeling read applications is complicated by the fact that the applications can select among several alternative sites for accessing fragments. For determining the redundant allocation of fragments, either of the following two methods can be used:
    1. Determine the set of all sites where the benefit of allocating one copy of the fragment is higher than the cost.
    2. For non replicated problem, introduce replicated copies starting from the most beneficial.

39 Data Replication

The data replication is mean replicated a relation **r** by copy it and stored in two or more or all sites.

Fully replication: in which a copy is stored in every site in the system.

there are many number of a advantage and disadvantage to replication :

Advantage:

1- Availability

If one of the sites containing relation **r** fails, then the relation **r** many be found in another site .thus the system may continue in process queries involving r despite the failure of one site.

2- Increased parallelism

Several sites can process queries involving **r** in parallel.

3- Data replications minimize movement of data between sites.

4- Replication enhances the performance of read operation and increase the availability of data to read transaction.

Disadvantage :

1- Increased overhead on update since the update must be propagated to all sites.

2- Problem of controlling concurrent updates by several transactions.

3- Management of replicas of relation r by choosing one of them as the primary copy of r.

40- Transparency and autonomy

A system can hide the details of the distribution of data in the network we call that (network transparency). The Network transparency is related to local autonomy

Network transparency: is the degree to which system users may remain unaware of the details of the design of the distributed system.

Local Autonomy: is the degree to which designer or administrator of one site may be independent of the remainder of the distributed system.

We shall consider the issues of transparency and autonomy from the points of view of:

- Naming of data items.
- Replication of data items
- Fragmentation of data items
- Location of fragments and replicas.

41- Naming of data items

Every data items must have unique name in database, this property is easy to ensure in anon distributed data base, but in distributed database care must be taken to ensure that two sites do not use the same name for distinct data items, May solution to this problem is:

1- Registered in a central name server all the items name.
but this solution has many disadvantage like :

- The name server may become a bottleneck.
- If name server crashes, any site in distributed system can not continue to run .
- There is little local autonomy since naming is controlled centrally.

2- Each site prefix its own site identifier to any name it generated. this approach increaser local autonomy and no central control required but the disadvantage of this approach is fails to achieve network transparency since identifiers are attached too names as site 17.deposit each replica of data item and each fragment of a data item must have unique name .it important that the system be able to determine those replicas that are replicas of same data item and those fragments that are fragments of the same data item .so we adopt the convention of post filing "f1","f2",……,"fn" to fragments of data item and "r1","r2",….."r n" to replicas thus *Site17.deposit.f3.r2.* Refers to replica 2 of fragment 3 of deposit, and this item was generated by sit 17. Each replica of data item and each fragment of data item must have a unique name.

42 Replication and fragmentation transparency

In replication, the users should not know a specific replica of data item and the system should:

1- Determine which replica to reference on a read request.
2- Update all replicas on write request so a catalog table is used by the system to determine all replicas for the data item.

In fragmentation, the user should not be required to know how a data item is fragments since:

1- Vertical fragments may contain tuple_ids which represent addresses of typle.
2- Horizontal fragments may involve complicated selection predicates.

So DDBS should allow request to be stated in terms of the un fragmented data items as well as it is always possible to reconstruct the original data item from its fragments.

43 Location Transparency

If replication and fragmentation transparency are provided by the system a large part of the DDB is hidden from the users but the site identifier component of names force the user to be ware of the fact that the system is distributed.

Location transparency is achieved by creating a set of alternative names or aliases for each user. a user may thus refer to data items by simple names that are translated by the system to complete names with aliases:

- The user can be unaware of the physical location of data item.
- The user is unaffected if the DBA decide to move a data item from one site to another.

44- Distributed query processing

An access operation issued by an application can be expressed as a query which references global relations. the DDBMS has to transform this query into simpler queries which refer only to fragments.

In general there is several different ways to transform a query over global relations called (global query) into queries over fragments called (fragment queries). These different transformation produce equivalent fragment queries which they produce the same result. There is many rules that can be applied to a query in order to rewrite it into an equivalent expression. There is variety of methods for computing the answer to query. The important measurement for processing a query that minimize the amount of time it takes to compute the answer in centralized system s the primary criterion for measuring the cost of particular strategy is the number

of disk accesses but in distributed system there are many points must take into our account :

- The cost of data transmission over the network.
- The potential gain in performance from having several sites process parts of the query in parallel.
- The cost of data transfer to and from disk in general we cannot focus on disk cost or on network cost we must find good tradeoff between them.

45- Recovery

Database recovery, it is a mechanics for restoring a database quickly and accurately after loss or damage.

Database are damaged or lost because:

- Some system problems that may be caused by human error.
- Hardware failure.
- Invalid or incorrect data.
- Program error.
- Computer viruses
- Natural catastrophes.

45-1 Why Recovery

The recovery is very important when one of the following problems be occurring:

1- System failure.
2- Procedures for periodically backup DB.
3- Restoring the database to usable state.
4- Re applying all lost transaction.

45-2 A recovery technique approaches:

1-Switch

In order to be switch to an existing copy of the database, at least two copies of database must be kept and update simultaneously. When failures occur processing is switching to the duplicate copy of the database this approach is faster recovery.

2-Restore/Rerun

This technique involves reprocessing the day's transaction (up to point failure) against the backup copy of the database. **The advantage of (restore/rerun):** The DBMS does not need to create a database change and no special restart procedures are required. **The disadvantage of (restore/rerun):**

- Several hours of reprocessing may be required.
- Processing new transaction will have to be deferred until recovery is complete

3-Transaction integrity

A database is update by processing transaction that result in changes to one or more database records if an error occurs during the processing of transaction the database may be compromised and some of database recovery is required.

4-Backup Recovery (rollback)

Used to back out undo unwanted changes to the database before images of the records that have been changed are applied to the database and the data base is returned to an earlier state used to reverse the changes made by transaction that have been a borated or terminated abnormally.

5-Forward recovery (roll forward)

Starts with an earlier copy of database after images (the image of good transaction) are applied to the database and the database is quickly moved forward to later state.

46-Recovery in distributed system

We can now consider the recovery problem in a distributed database. We assume that at each file local transaction manager is available which aid to local recovery .the basic technique for implementing transactions in presence of failures is based on the idea of undo. To undo the actions of a transaction of means to reconstruct the database as prior to its execution. To redo the actions of a transaction means to perform again its actions. The information of undo and redo are found in a log this process is easy in centralized database since transactions are atomic but in DDB it becomes much complicated to ensure the atomicity property of a transaction, since several sites may be participating in its execution. The failure of one of these sites or the failure of a communication link connecting these sites may result in erroneous computations. The function of transaction manager distributed database system is to ensure that the execution of the various transactions in the distributed system preserves atomicity. Each site has its own local transaction manager .the various transaction managers cooperate to execute global transactions.

Each site of the system contains two sub systems:

1- **Transaction manager** whose function is to manage the execution of those transactions (or sub transaction) that access data stored in local site. Each such

transaction may be either a local transaction (that is, a transaction that only executes at that site) or part of global transaction (that is a transaction that execute at several sites).

2- **Transaction coordinator** whose function is to coordinate the execution of the various transactions (both local and global) initiated at that site.

Each transaction manager is responsible for:

1- maintaining a log for recovery purposes.

2- participating in an appropriate Concurrency control scheme to coordinate the concurrency execution of the transaction execution at that site.

The transaction coordinator is not need in the centralized system since a transaction accesses data only at one single site.

Each Coordinator responsibility:

1- Starting the execution of the transaction.

2- Breaking the transaction in to a number of sub transactions and distributing these sub transaction.

3- To the proper ate sites for execution.

4- Coordinating the termination of the transaction, this may result in the transaction being committed at all sites or aborted at all sites.

47- Robustness

Distributed system may suffer from the same type of failure in a centralized like memory failure, disk crash in addition of that the distributed system including:

1- The failure of a site.

2- The failure of a link.

3- Loss of message.

4- Network partition.

In order for the system to be robustness it must detect any of these failures but in sometime it's difficult to distinguish the failure is it in a site or link or message. The system can usually detected that failure has occurred but it may not be able to identify what kind of failure it is.

Example: suppose that site S1 has discovered that a failure has occurred. what we do? It must initiate a procedure that will allow the system to reconfigure and continue with its normal mode of operation and take the following points with our account:

1. Replicated data stored at the failed site. The catalog should be updated so that queries do not reference the copy at the failed site.
2. If transaction were active at the failed site at the time of the failure these transaction should be aborted.
3. If the failure site is a central for some sub system.

   An election must be held to determine the new sever include a name server, concurrency, coordinator or global dead lock detector.

   Since it is in general not possible to distinguish between network link failures and site failures, any reconfiguration schema must be designed to work correctly in case of partitioning of the network.

   In particular the following situation must be avoided:

   1. Two or more central servers are elected in distinct partitions.
   2. More than one partition updates a replicated data item.

   The repaired site or link into the system also requires some care. When failed site recovers it must:

   1. Initiate a procedure to update its system table to reflect changes made while it was down.
   2. If site has replicas of any data items it must obtain the current values of these data
   3. Ensure that it receives all future update.

   Any easy solution is temporarily to halt the entire system while the failed site rejoins it. This temporary halt is unacceptably, the preferable solution is to represent the recovery tasks a series of transaction.

➢ *Strategic Information*

They need information to formulate the business strategies, establish goals, set objectives, and monitor results.
Here are some examples of business objectives:

- Retain the present customer base.
- Increase the customer base by 15% over the next 5 year.
- Enhance customer service level in shipments.
- Bring three new products to market in 2 years.
- Increase sales by 15% in the North East Division.

For making decisions about these objectives, managers need information for the following purposes: to get in-depth knowledge of their company's operations; learn about the key business factors and how these affect one another; monitor how the business factors change over time; and compare their company's performance relative to the competition and to industry benchmarks. managers need to focus their attention on customers' needs and preferences, emerging technologies, sales and marketing results, and quality levels of products and services. The types of information needed to make decisions in the formulation and execution of business strategies and objectives are broad-based and encompass the entire organization. We may combine all these types of essential information into one group and call it strategic information.

A producer wants to know

- ***What is the Data Warehouse?***

A data warehouse is a relational database that is designed for query and analysis rather than for transaction processing. It usually contains historical data derived from transaction data, but it can include data from other sources. In addition to a relational database, a data warehouse environment can include an Extraction, Transportation, Transformation, and Loading (ETL) solution, Online Analytical Processing (OLAP) and Data Mining capabilities, Client Analysis Tools , and other applications that manage the process of gathering data and delivering it to business users .

Formal Definition: “A data warehouse is a subject-oriented, integrated, time variant and non-volatile collection of data in support of management decision making process.”

It means:

• **Subject-Oriented**: *Stored data targets specific subjects.*
Example: It may store data regarding total Sales, Number of Customers, etc. and not general data on everyday operations.
Data warehouses are designed to help you analyze data. For example, to learn more about your company's sales data, you can build a data warehouse that concentrates on sales. Using this data warehouse, you can answer questions such as "Who was our best customer for this item last year?".

• **Integrated**: *Data may be distributed across heterogeneous sources which have to be integrated*. Example: Sales data may be on RDB, Customer information on Flat files, etc.
Integration is the most important. Data is fed from multiple disparate sources into the data warehouse. As the data is fed, it is converted, reformatted, re-sequenced, summarized, and so forth. The result is that data once it resides in the data warehouse has a single physical corporate image .

• **Time Variant***: Data stored may not be current but varies with time and data have an element of time*. Example: Data of sales in last 5 years, etc.
A data warehouse's focus on change over time is what is meant by the term time variant. In order to discover trends in business, analysts need large amounts of data.

• **Non-Volatile**: *The data warehouse is read-only; it generally has only 2 operations performed on it: Loading of data and Access of data*.
Nonvolatile means that, once entered into the data warehouse, data should not change. This is logical because the purpose of a data warehouse is to enable you to analyze what has occurred.

Most companies have realized that collecting transactional data is useful. In fact, it is tough to find any company that does not record their transactions. The data that has been collected for a number of years reside in various data sources—some in the mainframes, some in proprietary systems, and some in

client-server applications. Also, each of these systems was probably built and is being maintained by different people.

The typical dilemma of today's IT managers is not how to collect the data, but how to use the data accumulated over the years. The answer might sound simple: Put everything in one place and run reports against that database. Well, the programmer who built the mainframe system left the company 10 years ago. The consultants that were hired to build the proprietary system have since moved on to other jobs as well. Finally, you're already running the reports against the client server system you use for daily data collection, but those reports are fairly rigid—after they're printed, you can't really change or customize them. Each time you need a specific report, you have to pay a premium rate for a week or two to the outside consultant or to your own programmer. What can you do?

The goal of a data warehouse is to provide your company with an easy and quick look at its historical data. Advanced OLAP (on-line analytical processing) tools let DW users generate reports at a click of a mouse and look at the company's performance from various angles. How much data you need to examine depends on the nature of your business.

Suppose you have a manufacturing plant that produces thousands of parts per hour. The type of information you might be interested in includes the number of defects per hour or per day. Although you might want to examine the number of defective parts this year against the same number five years ago, such a ratio probably wouldn't provide the best picture of the company's performance. On the other hand, if you're in a car rental business, you might want to examine the number of customers this month against the same number six months ago. If you need to analyze the purchasing trends for customers with various demographic backgrounds, you might wish to examine data collected for a number of years. In short, if you need to make use of the data residing in some or all of your systems, you need to build a data warehouse.

- ***What is the different between operations of Data base systems and data warehouse?***

	data warehouse	**Operational Database**
User	Knowledge worker	Clerk
Function	Decision support	Day to day operation
Data	Historical ,Summarized Multidimensional, Integrated	Current, up-to-date, detailed
Unit of Work	Complex query	Short, Simple transaction

The major distinguishing features between OLTP and OLAP are summarized as follows:

- **OLTP (on-line Transaction processing)**
  - Major task of traditional relational DBMS.
  - Day-to-day operations: purchasing, inventory, banking, payroll, registration, accounting, etc.
- **OLAP (on-line Analytical processing)**
  - Data analysis and decision making (major task of data warehouse system).

Distinct features	**OLTP**	**OLAP**
User and system orientation	customer	market
Data contents	current, detailed	historical
Database design	**ER**(entity-relationship)data module+ **application**	star + subject
View	current, local	evolutionary, integrated
Access patterns	update	read-only but complex querie

- ***OLTP Systems are used to "run" a business.***
- ***The Data Warehouse helps to "optimize" the business.***

➢ *Why Have a Separate Data Warehouse?*

Three Main reasons:

1. OLTP systems require high concurrency, reliability, locking which provide good performance for short and simple OLTP queries. An OLAP query is very complex and does not require these properties. Use of OLAP query on OLTP system degrades its performance.

2. An OLAP query reads HUGE amount of data and generates the required result. The query is very complex too.

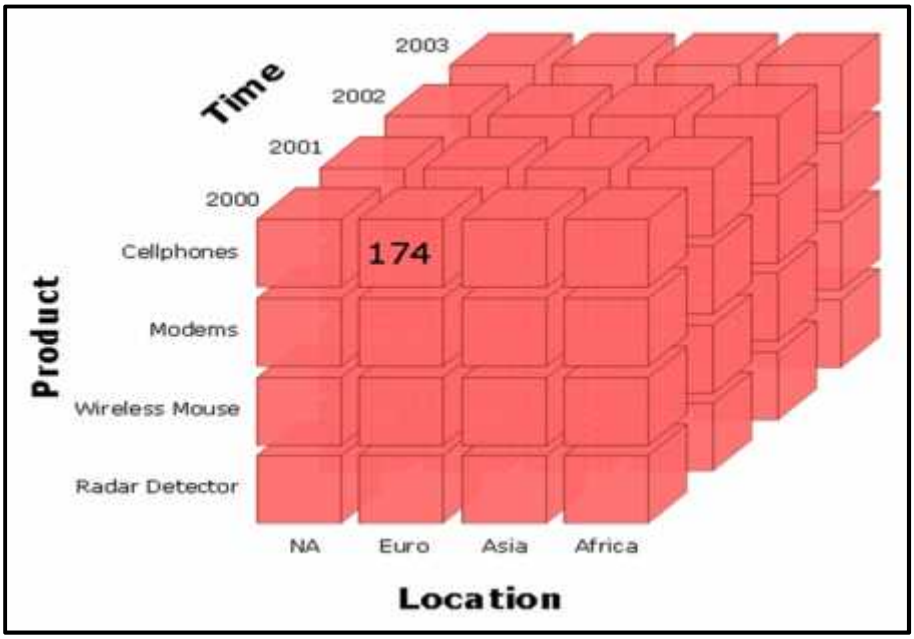
3. OLAP systems access historical data and not current volatile data while OLTP systems access current up-to-date data and do not need historical data.

Thus, Solution is to have a separate database system which supports primitives and structures suitable to store, access and process OLAP specific data … in short…have a data warehouse.

➢ Multidimensional data model

Data warehouse systems provide online analytical processing (OLAP) tools for interactive analysis of multidimensional data at varied levels. OLAP tools typically use the data cube and a multidimensional data model to provide flexible access to summarized data.

- **Data cube**
  - Data cube is a structure that enables OLAP to achieve the multidimensional functionality.
  - The data cube is used to represent data along some measure of interest.
  - Data Cubes are an easy way to look at the data (allow us to look at complex data in a simple format).
  - Although called a "cube", it can be 2-dimensional, 3-dimensional, or higher-dimensional.
  - Databases designs are for OLTP and efficiency in data storage.
  - Data cube design is for efficiency in data retrieval (ensures report optimization).
  - The cube is comparable to a table in a relational database.



Dimensions and Measures

- **Dimensions and Measures**

Data cubes have categories of data called dimensions and measures.

- **Measure**

Represents some fact (or number) such as cost or units of service.

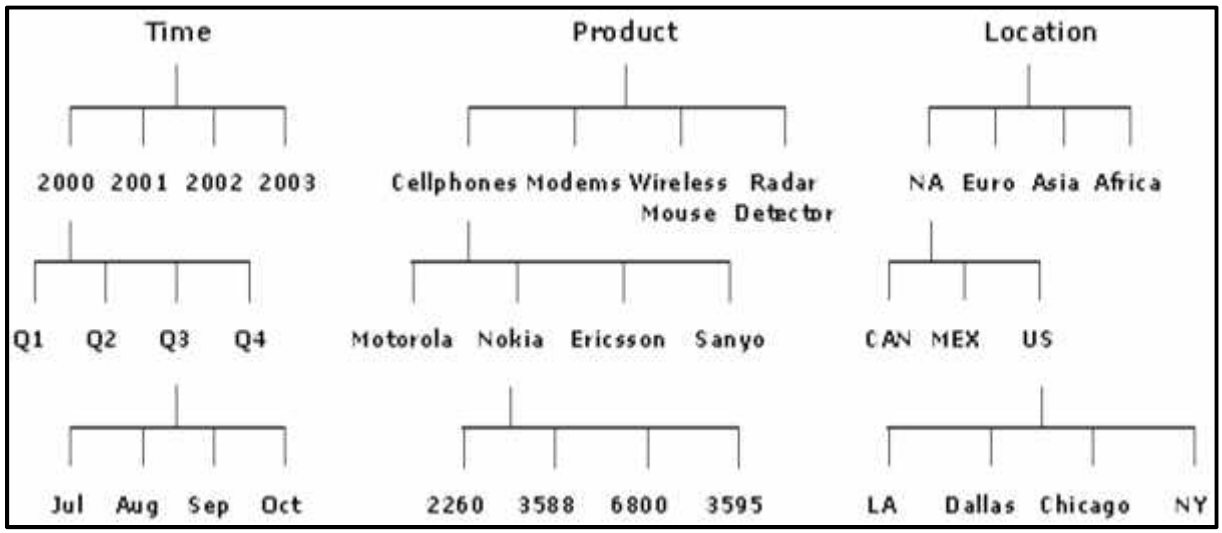
- The measures are the actual data values that occupy the cells as defined by the dimensions selected.
- Measures include facts or variables typically stored as numerical fields.

Volume of Product (Numbers in 1000)	CellPhones	Modems	Wireless Mouse	Radar Detector
2003	101	84	98	75
2002	94	60	70	60
2001	88	65	80	85
2000	105	82	88	93

- **Dimension**

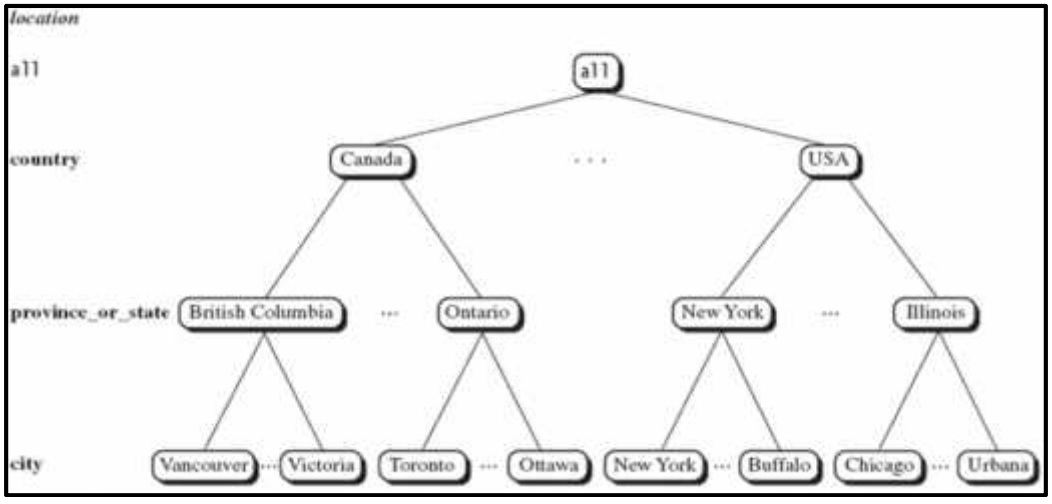
  Represents descriptive categories of data such as time or location**.**

  Each dimension includes different levels of categories**.**



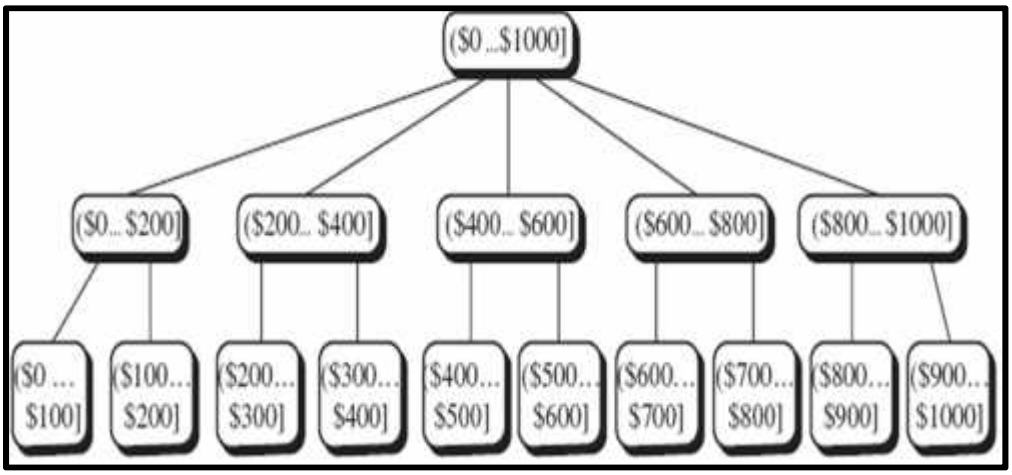
*Dimensions are organized into **hierarchies***

- E.g., Time dimension: days → weeks → quarters
- E.g., Product dimension: product → product line → brand
- A Concept **Hierarchy** defines *a sequence of mappings from a set of low-level concepts to high-level.*
- Consider a concept hierarchy for the dimension “Location”



Concept Hierarchies

Concept hierarchies may also be defined by grouping values for a given dimension or attribute, resulting in a set-grouping hierarchy



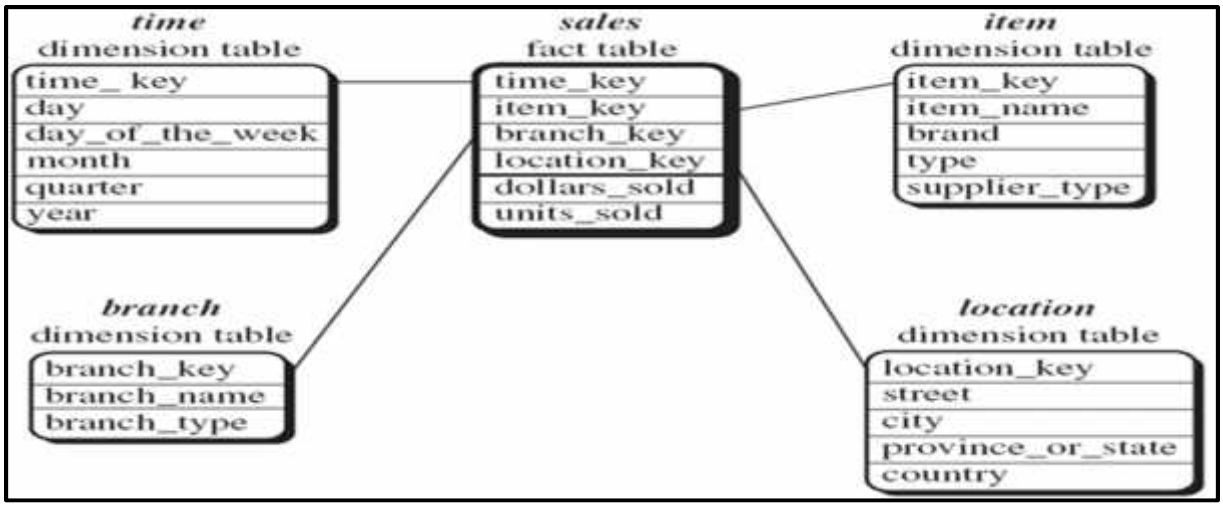
•Schemas for Multidimensional Databases

A data warehouse requires a concise, subject-oriented schema that facilitates on-line data analysis. The most popular data model for a data warehouse is a multidimensional model. Such a model can exist in the form of a **star schema, a snowflake schema, or a fact constellation schema**. Let's look at each of these schema types.

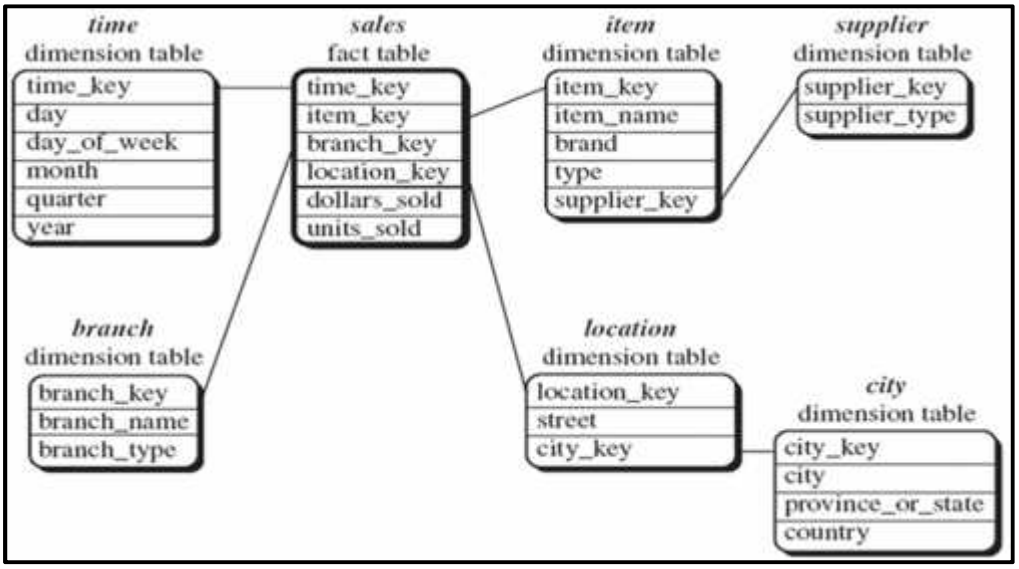
1. **Star schema**: A fact table in the middle connected to a set of dimension tables

It contains:

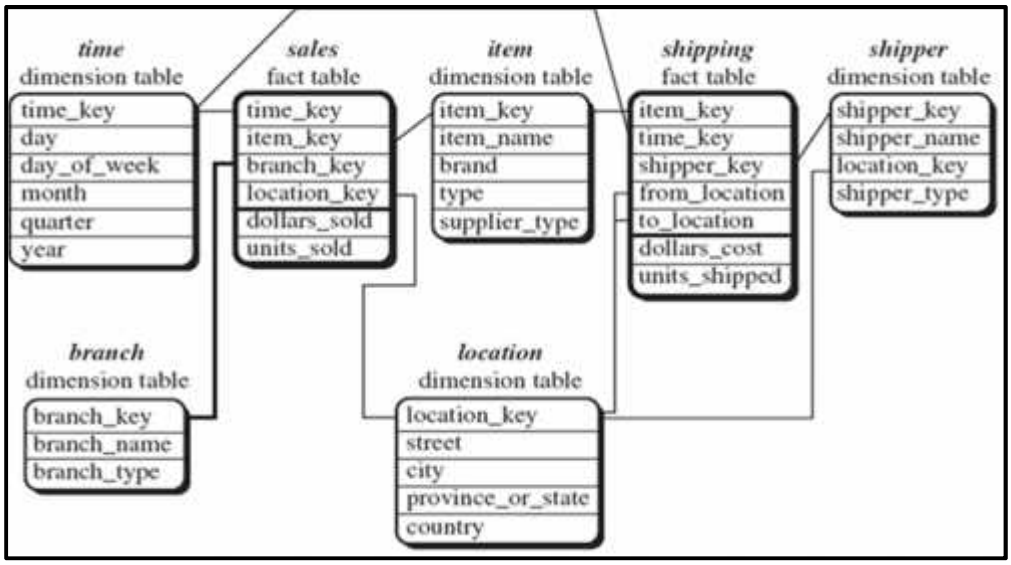
- A large central table (fact table)
- A set of smaller tables (dimension table), one for each dimension



2. **Snow flake schema**: A refinement of star schema where some dimensional hierarchy is further splitting (normalized) into a set of smaller dimension tables, forming a shape similar to snowflake, However, the snowflake structure can reduce the effectiveness of browsing, since more joins will be needed



3. **Fact constellations**: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation


time
dimension table
time_key
day
day_of_week
month
quarter
year
sales
fact table
time_key
item_key
branch_key
location_key
dollars_sold
units_sold
item
dimension table
item_key
item_name
brand
type
supplier_type
shipping
fact table
item_key
time_key
shipper_key
from_location
to_location
dollars_cost
units_shipped
shipper
dimension table
shipper_key
shipper_name
location_key
shipper_type
branch
dimension table
branch_key
branch_name
branch_type
location
dimension table
location_key
street
city
province_or_state
country

- **On line analysis type (OLAP)**

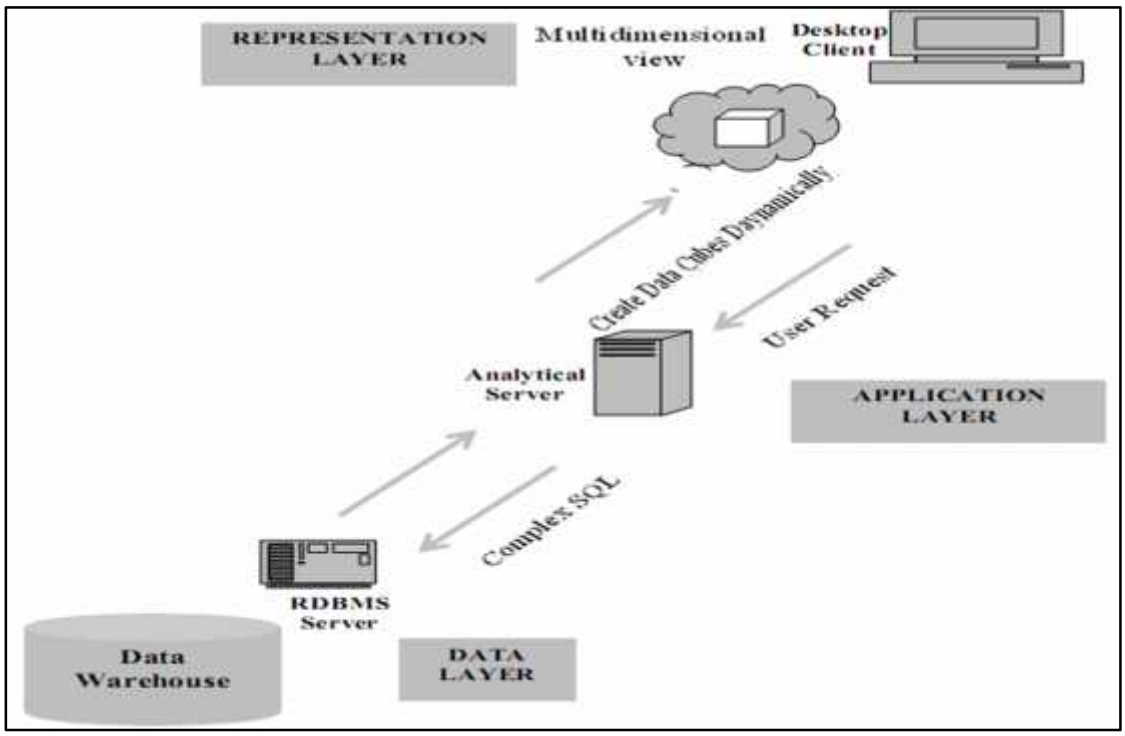
OLAP uses a Snapshot of a database taken at one point in time and then puts the data into a dimensional model. The purpose of this model is to run queries that deal with aggregations of data rather than individual transactions. Data warehousing and On-Line Analytical Processing (OLAP) are essential elements of decision support, which has increasingly become a focus of the database industry. Many commercial products and services are now available, and all of the principal database management system vendors now have offerings in these areas. Decision support places some rather different requirements on database technology compared to traditional On-Line Transaction Processing (OLTP) applications. OLAP tools allow users to make ad hoc queries or generate canned queries against the warehouse database. The OLAP category has been divided further into the multidimensional OLAP (MOLAP) and relational OLAP (ROLAP) markets.

Data warehouses and OLAP are necessary elements of Decision Support Systems (DSSs). They enable business decision makers to creatively approach, analyze and understand business problems. While data warehouses are built to store very large amounts of integrated data used to assist the decision-making process, the concept of OLAP, which is first formulated in 1993 to enable business decision makers to work with data warehouses, supports dynamic synthesis, analysis, and consolidation of large volumes of multidimensional data.

- **Categorization of OLAP**

1. **ROLAP (Relational OLAP**)

This type uses relational databases (RDMS) to store the data, sometimes by using a star schema or snowflake schema. ROLAP tools present sophisticated SQL and navigational methods on top of traditional relational databases. Relational OLAP tools present scalable, manageable technologies for very large data. ROLAP databases can easily handle dimensions with high cardinality. Figure shows the architecture of the ROLAP model. What you see is three-tier architecture.



The architecture of the ROLAP

Advantages:

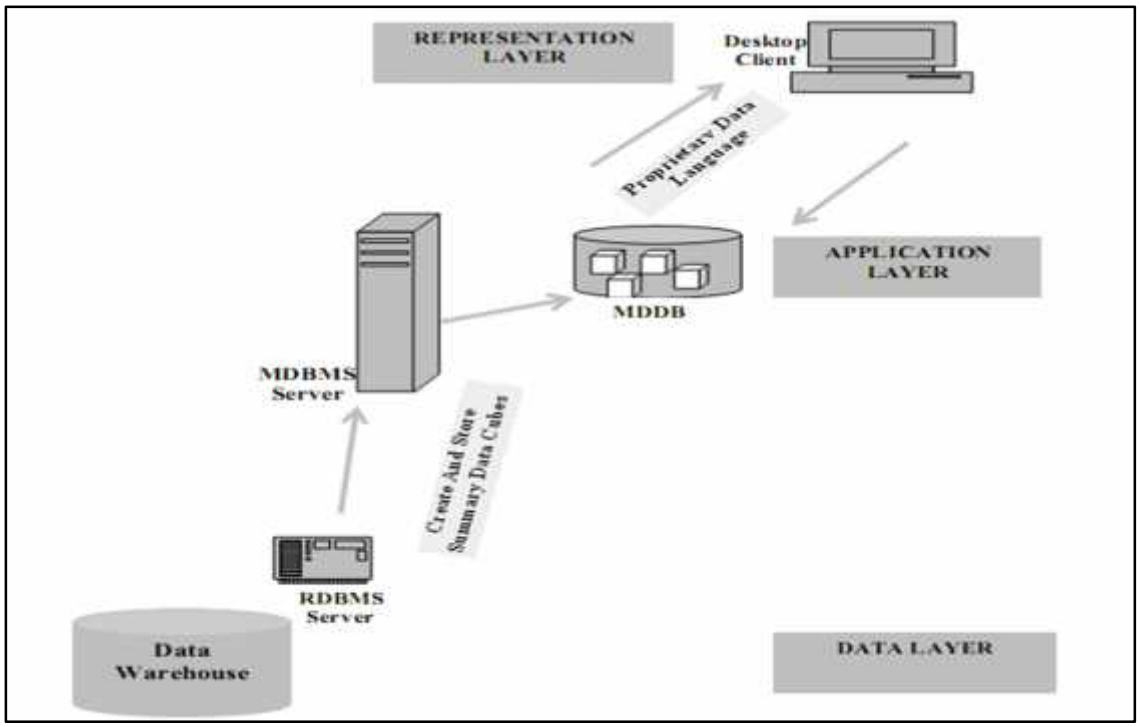
- It can handle large amounts of data.
- It can leverage functionalities inherent in the relational database.

Disadvantage:

- Performance can be slow because each ROLAP report is essentially a SQL query (or multiple SQL queries) in the relational database.
- It is limited by SQL functionalities because ROLAP technology mainly relies on generating SQL statements to query the relational database, and SQL statements do not fit all needs.

2. **MOLAP (Multidimensional OLAP):**

In the MOLAP, data is extracted from the data warehouse and aggregated into a data structure, commonly referred to as a cube, for analysis. Uses a specialized data store with pre_aggregated summaries to store the data. The MOLAP data store is built specifically to handle multidimensional queries and offers fast, efficient, and manageable access to multidimensional data. Figure shows the architecture of the MOLAP model.



Architecture of the MOLAP model

Advantages:

- It has excellent performance: A MOLAP cube is built for fast data retrieval, and is optimal for slicing and dicing operations.
- It can perform complex calculations.

Disadvantage:

- Limited in the amount of data it can handle because all calculations are performed when the cube is built
- It requires additional investment: Cube technologies is often proprietary and do not already exists in the organization.

3. HOLAP (Hybrid OLAP):

It bridges the technology gap between ROLAP and MOLAP, enabling you to use both multidimensional data stores (MDDB) and RDBMS data stores.

➢ Data warehouse architecture

• A Three-Tier Data Warehouse Architecture

Data warehouses often adopt three-tier architecture, as presented in the following Figure:

1- **Bottom tier** is a warehouse database server that is almost always a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external sources. These tools and utilities perform data extraction, cleaning, and transformation.

as well as load and refresh functions to update the data warehouse . The data are extracted using application program interfaces known as **gateways**. A gateway is supported by the underlying DBMS and allows client programs to generate SQL code to be executed at a server.

This tier also contains **a metadata repository**, which stores information about the data warehouse and its contents.

A **metadata repository** is data about data. When used in a data warehouse, metadata are the data that define warehouse objects.

A metadata repository should contain the following:

- A description of the structure of the data warehouse.
- Operational metadata.
- The algorithms used for summarization.
- The mapping from the operational environment.
- Data related to system performance.
- Business metadata.

2. **Middle tier** is an **OLAP server** that is typically implemented using either

(1) A relational OLAP (ROLAP) model.

(2) A multidimensional OIAP (IOLAP) model.

★ **OLAP server** Logically, OLAP servers present business users with multidimensional data from data warehouses or data marts, without concerns regarding how or where the data are stored. However, the physical architecture and implementation of OLAP servers must consider data storage issues

2. **Top tier** is a *front-end client layer*, which contains query and reporting tools, analysis tools, and/or data mining tools.

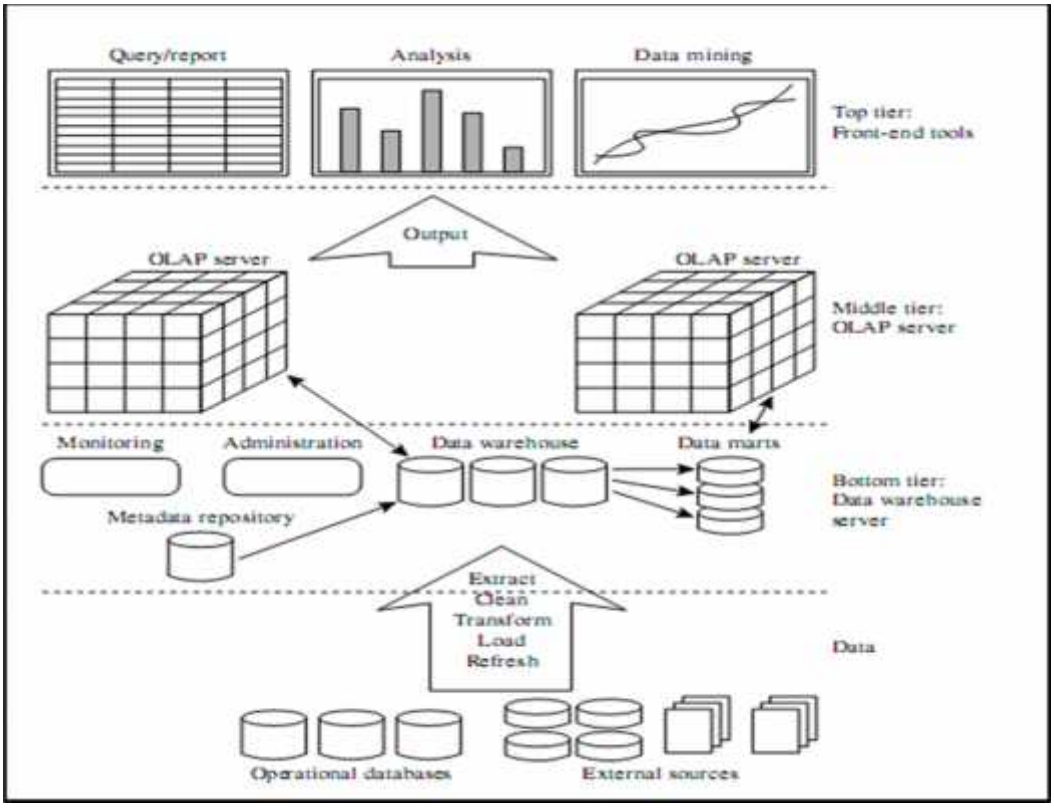


Figure () : A three-tier data warehousing architecture

❖ Business Analysis Framework

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 .

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.

❖ Data Warehousing vs. Data Marts

Data warehousing and data mart are tools used in data storage. With passage of time, small companies become big, and this is when they realize that they have amassed huge amounts of data in various departments of the organization. Every department has its own database that works well for that department. But when organizations intend to sort data from various departments for sales, marketing or making plans for future, the process is referred to as Data Mining. Data Warehousing and Data Marts are two tools that help companies in this regard. Just what the difference between data warehousing and data marts is and how they compare with each other is what this article intends to explain.

- **Data Warehousing**

This is the place where all the data of a company is stored. It is actually a very fast computer system having a large storage capacity. It contains data from all the departments of the company where it is constantly updated to delete redundant data. This tool can answer all complex queries pertaining data.

- **Data Mart**

It is an indexing and extraction system. Instead of putting the data from all the departments of a company into a warehouse, data mart contains database of separate departments and can come up with information using multiple databases when asked.

IT managers of any growing company are always confused as to whether they should make use of data marts or instead switch over to the more complex and more expensive data warehousing. These tools are easily available in the market, but pose a dilemma to IT managers.

- **Difference between Data Warehousing and Data Mart**

It is important to note that there are huge differences between these two tools though they may serve same purpose. Firstly, data mart contains programs, data, software and hardware of a specific department of a company. There can be separate data marts for finance, sales, production or marketing. All these data marts are different but they can be

coordinated. Data mart of one department is different from data mart of another department, and though indexed, this system is not suitable for a huge data base as it is designed to meet the requirements of a particular department.

Data Warehousing is not limited to a particular department and it represents the database of a complete organization. The data stored in data warehouse is more detailed though indexing is light as it has to store huge amounts of information. It is also difficult to manage and takes a long time to process. It implies then that data marts are quick and easy to use, as they make use of small amounts of data. Data warehousing is also more expensive because of the same reason.

Summary

• Data mart and data warehousing are tools to assist management to come up with relevant information about the organization at any point of time

• While data marts are limited for use of a department only, data warehousing applies to an entire organization

• Data marts are easy to design and use while data warehousing is complex and difficult to manage

• Data warehousing is more useful as it can come up with information from any department

❖ Top-down versus bottom-up design methodologies

Bottom-up design

Created an approach to data warehouse design known as bottom-up. In the bottom-up approach, data marts are first created to provide reporting and analytical capabilities for specific processes. These data marts can eventually be integrated to create a comprehensive data warehouse. The data warehouse bus architecture is primarily an implementation of "the bus", a collection of conformed dimensions and conformed facts, which are dimensions that are shared (in a specific way) between facts in two or more data marts.

Top-down design

Data warehouse is a centralized repository for the entire enterprise. The top-down approach is designed using a normalized enterprise data model. "Atomic" data, that is, data at the lowest level of detail, are stored in the data warehouse. Dimensional data marts containing data needed for specific business processes or specific departments are created from the data warehouse. The data warehouse is at the center of the "Corporate Information Factory" (CIF), which provides a logical framework for delivering business intelligence (BI) and business management capabilities.

- **Steps of design and construction of data warehouse**

In general, the warehouse design process consists of the following steps:

1. Choose a business process to model (e.g., orders, account administration, sales, or the general ledger).

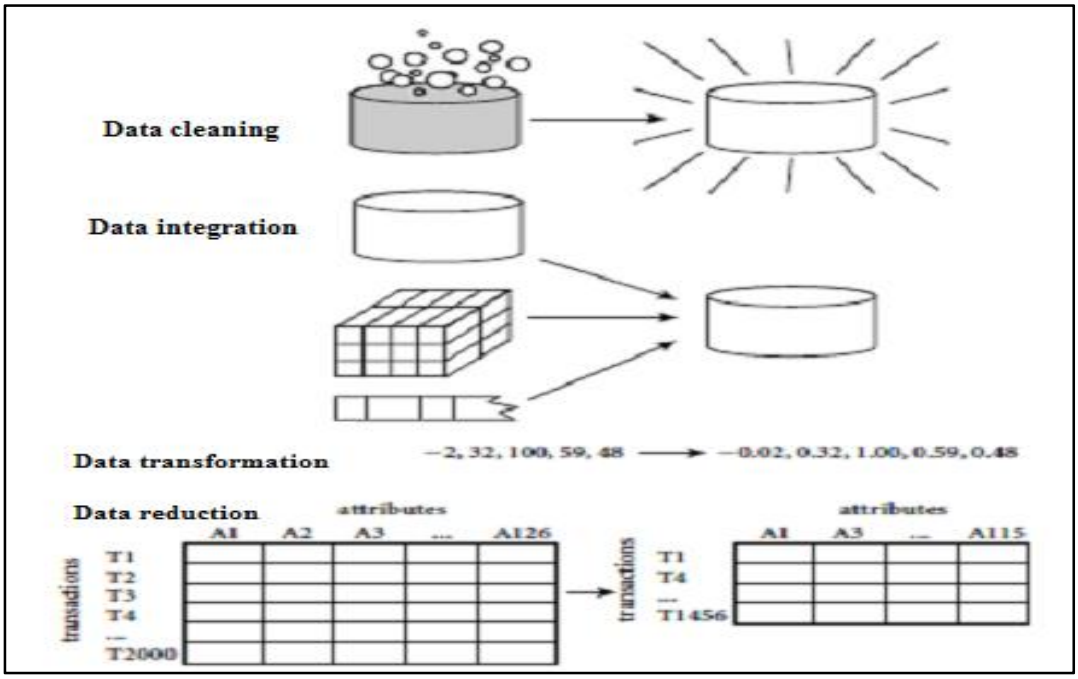
2. Choose the business process, which is the fundamental, atomic level of data to be represented in the fact table for this process (e.g., individual transactions, individual daily snapshots, and so on).

3. Choose the dimensions that will apply to each fact table record. Typical dimensions are time, item, customer, supplier, warehouse, transaction type, and status.

4. Choose the measures that will populate each fact table record. Typical measures are numeric additive quantities like dollars sold and units sold.

➢ *Data Preprocessing*

Major Tasks in Data Preprocessing



Forms of data preprocessing

- **Data cleaning**:
- **Data integration**
- **Data transformation**
- **Data Reduction**

2.1 Data cleaning:

Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies.

2.1.1 *Missing Values*

Lacking attribute values, lacking certain attributes of interest, or containing only aggregate data. e.g., Occupation=” ” (missing data)

How to Handle Missing Data?

1. **Ignore the tuple**: usually done when class label is missing (when doing classification)—not effective when the % of missing values per attribute varies considerably.

2. **Fill in the missing value manually**: this approach is time-consuming and may not be feasible given a large data set with many missing values.

3. **Use a global constant to fill in the missing value**: Replace all missing attribute values by the same constant, such as a label like "Unknown". If missing values are replaced by, say, "Unknown," then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common—that of "Unknown." Hence, although this method is simple, it is not foolproof.

4. **Use the attribute mean to fill in the missing value**: For example, suppose that the average income of All Electronics customers is $56,000. Use this value to replace the missing value for income.

5. **Use the attribute mean for all samples belonging to the same class as the given tuple:** For example: if classifying customers according to credit risk replace the missing value with the average income value for customers in the same credit risk category as that of the given tuple.

6. The most probable value: **inference-based such as Bayesian formula or decision tree.**

2.1.2 *Noisy Data*

Noise is a random error or variance in a measured variable. e.g., Salary="−10" (an error)

- **Data smoothing techniques**: In general

1- **Binning**

E.g.: Sorted data for price (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34

In this example, the data for price are first sorted and then partitioned into equal-frequency bins of size 3 (i.e., each bin contains three values)

Bin 1: 4, 8, 15

Bin 2: 21, 21, 24

Bin 3: 25, 28, 34

- **Smoothing by bin means**

Each value in a bin is replaced by the mean value of the bin.

e.g. (4+8+9)/3= 9, (21+21+ 24)/3=22, (25+ 28+ 34)/3=29

Bin 1: 9, 9, 9

Bin 2: 22, 22, 22

Bin 3: 29, 29, 29

- **smoothing by bin boundaries**

The minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value.

Smoothing by bin boundaries:

- Bin 1: 4, 4, 4, 15
- Bin 2: 21, 21, 25, 25
- Bin 3: 26, 26, 26, 34

2. **Regression**: smooth by fitting the data into regression functions.

3. **Clustering:** detect and remove outliers.

2.2 Data integration

Combines data from multiple sources into a coherent data store, as in data warehousing, these sources may include multiple databases, data cubes, or flat files.

Advantage: Careful integration of the data from multiple sources may help reduce/avoid redundancies and inconsistencies and improve mining speed and quality.

There are a number of issues to consider during data integration

Redundancy

Redundant attributes may be able to be detected by ***correlation analysis*** and ***covariance analysis***.

1. ***correlation analysis (Nominal Data):***

X^2 (chi-square) test

$$¡^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

The observed frequency (i.e., actual count) of the joint event

The **expected eij** frequency which can be computed as

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{N};$$

Where N is the number of data tuples, count(A=ai) is the number of tuples having value ai for A, and count(B = bj) is the number of tuples having value bj for B.2.3 Data transformation

Gender / preferred reading	male	female	Sum (row)
Like fiction	250	200	450
Not like fiction	50	1000	1050
Sum(col.)	300	1200	1500

Example 1.1 Correlation analysis of categorical attributes using c2. Suppose that a group of 1,500 people was surveyed. The gender of each person was noted. Each person was polled as to whether their preferred type of reading material was fiction or nonfiction. Thus, we have two attributes, gender and preferred reading

1. Expected frequencies

e11 = [count (male) x count (fiction)]/N

e11= [300 x 450] /1500= 90

e12= [1200 x 450]/1500=360

e21= [300 x 1050]/1500=210

e22== [1200 x 1050]/1500=360

Gender / preferred reading	Play chess	Not play chess	Sum (row)
Like science fiction	250(90)	200(360)	450
Not like science fiction	50(210)	1000(840)	1050
Sum(col.)	300	1200	1500

2. Compute X^2 $\quad i^2 = \sum \frac{(Observed - Expected)^2}{Expected}$

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840}$$
$$= 284.44 + 121.90 + 71.11 + 30.48 = 507.93.$$

It shows that ***preferred reading*** and **Gender** are correlated in the group

2.3 Data transformation

The data are transformed or consolidated into forms appropriate for mining.

Data transformation can involve the following:

a. **Smoothing**: which works to remove noise from the data Such techniques include binning, regression, and clustering.

b. **Aggregation:** Summarization, data cube construction.

c. **Generalization**: of the data, where low-level or "primitive" (raw) data are replaced by higher-level concepts through the use of concept hierarchies. For example, categorical attributes, like street, can be generalized to higher-level concepts, like city or country. Similarly, values for numerical attributes, like age, may be mapped to higher-level concepts, like youth, middle-aged, and senior.

d. **Attribute construction** (or feature construction): where new attributes are constructed and added from the given set of attributes to help the mining process.

e. Normalization: Scaled to fall within a smaller, specified range

1) **min-max normalization**
2) **z-score normalization**
3) **normalization by decimal scaling**

Advantage:

Normalization is particularly useful for classification algorithms involving **neural networks**, or distance

measurements such as **nearest-neighbor** classification and **clustering**.

1) **min-max normalization**
Min-max normalization performs a linear transformation on the original data. Suppose that $\boldsymbol{min_A}$ and $\boldsymbol{max_A}$ are the minimum and maximum values of an attribute, *A*. Min-max normalization maps a value, *v*, of *A* to *v′* in the range [$\boldsymbol{new\_min_A}$, $\boldsymbol{new\_max_A}$] by computing

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A.$$

Min-max normalization preserves the relationships among the original data values. It will encounter an "out-of-bounds" error if a future input case for normalization falls outside of the original data range for A.

Example 1.2
Min-max normalization. Suppose that the minimum and maximum values for the attribute income are $12,000 and $98,000, respectively. We would like to map income to the range [0:0;1:0]. By min-max normalization, a value of $73,600 for income is transformed to

$$\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716$$

2) **z-score normalization** (or zero-mean normalization), the values for an attribute, A, are normalized based on the mean and standard deviation of A. A value, v, of A is normalized to v′ by computing

$$v' = \frac{v - \bar{A}}{\sigma_A},$$

Where $\bar{A}$ and $\boldsymbol{\sigma_A}$ are the mean and standard deviation, respectively, of attribute A. This method of normalization is useful when the actual minimum and maximum of attribute A are unknown, or when there are outliers that dominate the min-max normalization.

Example 2.3
z-score normalization Suppose that the mean and standard deviation of the values for the attribute income are $54,000 and $16,000, respectively. With z-score normalization, a value of $73,600 for income is transformed to

$$\frac{73{,}600-54{,}000}{16{,}000} = 1.225.$$

3) **Normalization by decimal scaling** normalizes by moving the decimal point of values of attribute A. The number of decimal points moved depends on the maximum absolute value of A. A value, v, of A is normalized to v′ by computing

$$v' = \frac{v}{10^j},$$

Where j is the Max (|V|) < 1.

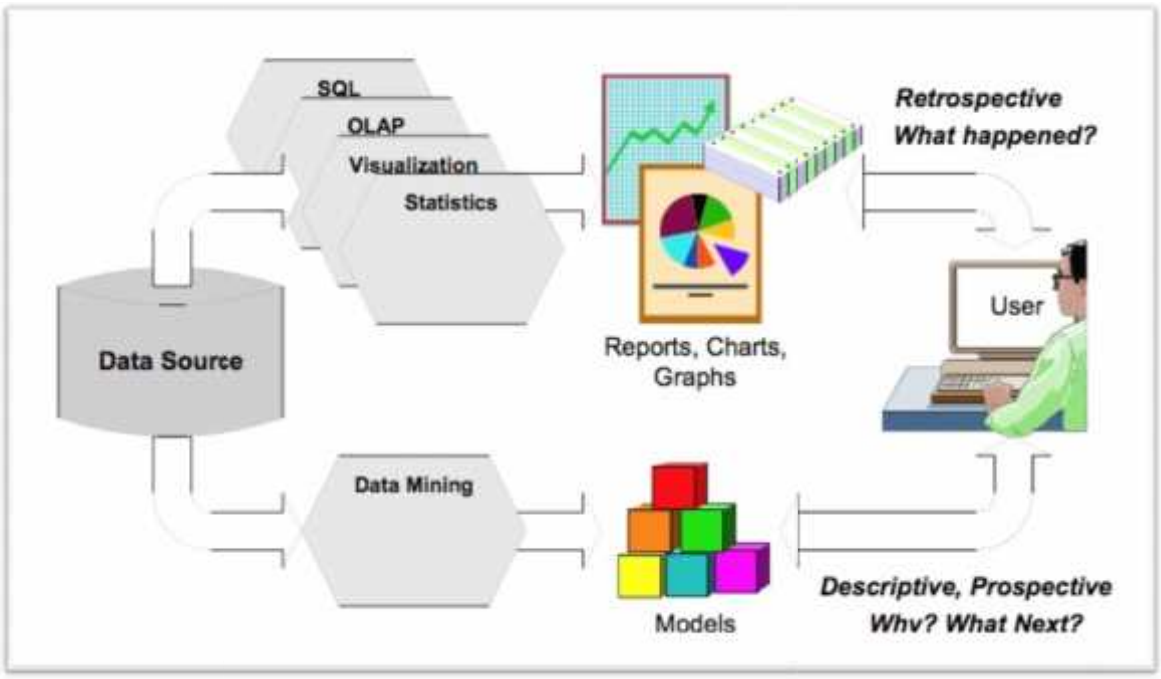
Example 1.4
Decimal scaling. Suppose that the recorded values of A range from -986 to 917. The maximum absolute value of A is 986. To normalize by decimal scaling, we therefore divide each value by 1,000 (i.e., j = 3) so that -986 normalizes to -0,986 and 917 normalizes to 0,917.

2.4 Data Reduction

Data reduction techniques can be applied to obtain a reduced representation of the data set that is much smaller in volume, yet closely maintains the integrity of the original data. That is, mining on the reduced data set should be more efficient yet produce the same (or almost the same) analytical results.

➢ *Data Mining*

Data mining is a new component in an enterprise's decision support system (DSS) architecture. It complements and interlocks with other DSS capabilities such as query and reporting, on-line analytical processing (OLAP), data visualization, and traditional statistical analysis. Shown in Figure bellow , these other DSS technologies are generally retrospective. They provide reports, tables, and graphs of what happened in the past. A user who knows what she's looking for can answer specific questions like: "How many new accounts were opened in the Midwest region last quarter," "Which stores had the largest change in revenues compared to the same month last year," or "Did we meet our goal of a ten-percent increase in holiday sales?"



Data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases.

- ***The Purpose of Data Mining :***

1. There is often information “hidden" in the data is not easily evident.

2. Human analysts may take weeks to discover useful information.

3. Much of the data is never analyzed at all.

- ***Data Mining Functionalities—What Kinds of Patterns Can Be Mined?***

The data mining functionalities and the kinds of patterns they can discover, are described Below:

1- **Characterization**: Data characterization is a summarization of general features of objects in a target class, and produces what is called characteristic rules. The data relevant to a user-specified class are normally retrieved by a database query and run through a summarization module to extract the essence of the data at different levels of abstractions. For example, one may want to characterize the Our Video Store customers who regularly rent more than 30 movies a year.

2-**Discrimination**: Data discrimination produces what are called discriminate rules and is basically the comparison of the general features of objects between two classes referred to as the target class and the contrasting class. For example, one may want to compare the general characteristics of the customers who rented more than 30 movies in the last year with those whose rental account is lower than 5. The techniques used for data discrimination are very similar to the techniques used for data characterization with the exception that data discrimination results include comparative measures.

3- **Association analysis**: Association analysis is the discovery of what are commonly called association rules. It studies the frequency of items occurring together in transactional databases. Association analysis is commonly used for market basket analysis. For example, it could be useful for the Our Video Store manager to know what movies are often rented

together or if there is a relationship between renting a certain type of movies and buying popcorn or pop. The discovered association rules are of the form: P->Q that P and Q appear together in a transaction, is the conditional probability that Q appears in a transaction when P is present.

4- **Classification**: Classification analysis is the organization of data in given classes. Also known as supervised classification. Classification approaches normally use a training set where all objects are already associated with known class labels. The classification algorithm learns from the training set and builds a model. The model is used to classify new objects. The derived model is based on the analysis of a set of training data "How is the derived model presented?" The derived model may be represented in various forms, such as classification (IF-THEN) rules, decision trees, mathematical formulae, or neural networks .

5- **Prediction**: the term prediction may refer to both numeric prediction and class label prediction ,Prediction is however more often referred to the forecast of missing numerical values, or increase/ decrease trends in time related data. The major idea is to use a large number of past values to consider probable future values.

6- **Clustering**: Similar to classification, clustering is the organization of data in classes. However, unlike classification, in clustering, class labels are unknown . Clustering is also called unsupervised classification, because the classification is not dictated by given class labels. There are many clustering approaches all based on the principle of maximizing the similarity between objects in a same class (intra-class similarity) and minimizing the similarity between objects of different classes (inter-class similarity).

7- **Outlier analysis**: Outliers are data elements that cannot be grouped in a given class or cluster. Also known as exceptions or surprises, they are often very important to identify. While outliers can be considered noise and discarded in some applications, they can reveal important knowledge in other domains, and thus can be very significant and their analysis valuable.