CHAPTER 5

UNIT V

Parallel and Distributed Databases

Syllabus

- Introduction to Database architectures: Multi-user DBMS architectures
- Case Study- Oracle Architecture
- **Parallel Databases :** Speedup and Scaleup, Architectures of Parallel Databases.
- **Distributed Databases :** Architecture of Distributed Databases, Distributed Database Design, Distributed Data Storage
- Distributed Transaction : Basics, Failure modes, Commit Protocols, Concurrency Control in Distributed Database

Syllabus Topic : Introduction To Database Architecture - Multi-user DBMS Architectures

5.1 Introduction to Database Architecture : Multi-user DBMS Architectures

The common architectures that are used to implement multi-user database management systems :

1. Teleprocessing
2. File-Server
3. Client-Server

5.1.1 Teleprocessing

- Teleprocessing is a traditional architecture for multiuser systems. This architecture has one computer with single Central Processing Unit(CPU) and multiple terminals. Here the processing is performed in one physical computer.
- The different terminal are typically "dumb", incapable of functioning on their own and cabled to the central computer.
- In this system, the terminals can access DBMS and database directly but this it also makes very heavy load to the CPU as it has to perform two operations
  - Run application programs and DBMS.
  - Formatting of data to represent on the terminals.

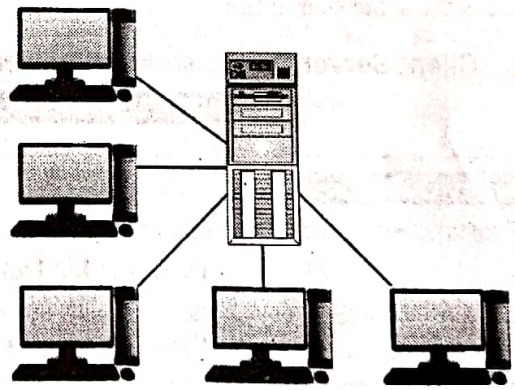


Fig. 5.1.1 : Teleprocessing architecture

5.1.2 File Server

- In the file-server architecture, there is a computer which is connected to a network and mainly serves as a shared storage.

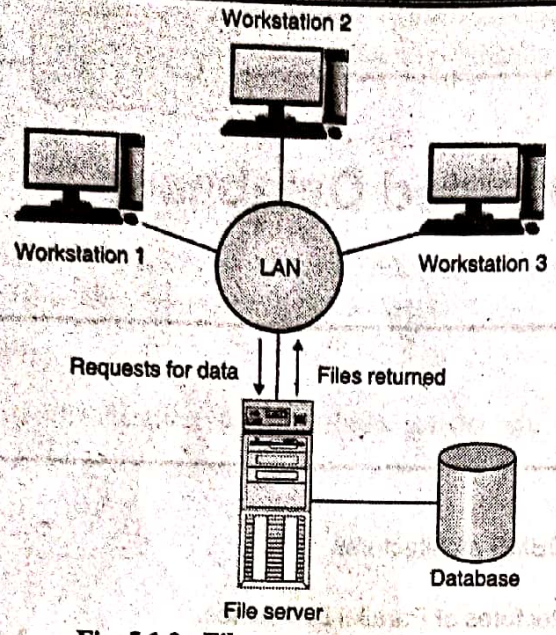


Fig. 5.1.2 : File server architecture

- The processing is distributed over the network in the file-server architecture is typically the Local Area Network (LAN). The file server stores all the files which are required to applications and DBMS. To get these files, the applications and DBMS has to make requests to file server.
- The file server works as shared data disc.

Disadvantages

- It generates heavy network traffic.
- Each workstation requires a full copy of DBMS.
- Complex integrity, concurrency, and recovery control : Multiple DBMSs may concurrently access the same shared file.

5.1.3 Client Server Database Architecture

SPPU - May 14, Dec. 15

University Questions

Q. Explain client server database architecture. **(May 2014, 3 Marks)**

Q. Explain Client Server Architecture with suitable database application. **(Dec. 2015, 5 Marks)**

- A network architecture in which each computer or process is connected in the network is client or server is known as **client server architecture.**
- Server is nothing but the powerful computer. It has more storage and processing capacity than the client machine. It manages client computers, printers, disk drivers and network traffic.
- Clients are the PC's where user can run the applications. Clients always depend on server for resources and processing power.
- In this architecture clients request data from the server and server provide all the required information to the client in response.

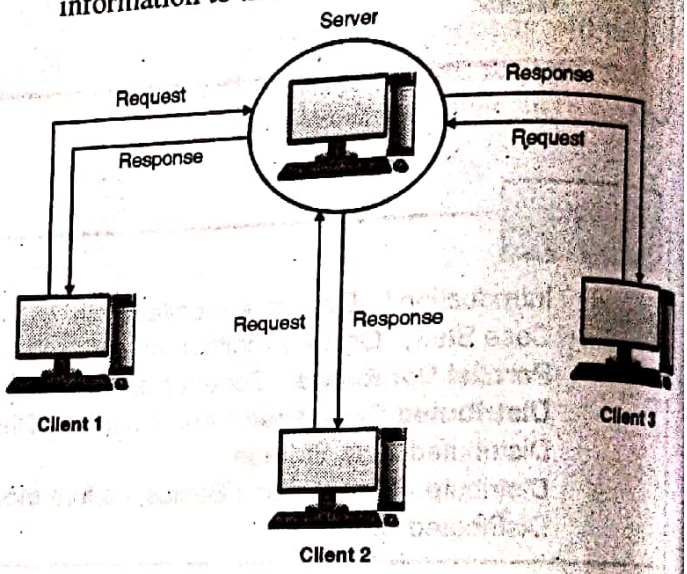


Fig. 5.1.3 : Client-server architecture

Banking Database Application of Client Server Architecture

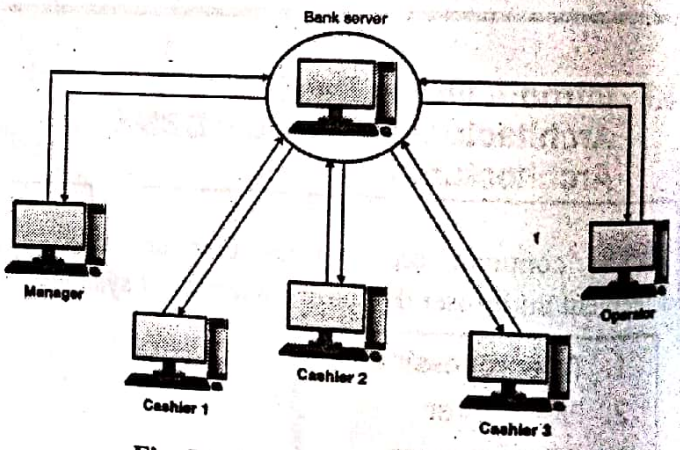


Fig. 5.1.4 : Bank database system

- Consider the Banking Database Example. In this system, the database is stored on server. From this server all clients like Manager, Operator and Cashiers in the system access the data from database. As per roles and privileges granted the data accessibility rights are decided for different clients in the bank.
- For example, the manager will have access right to whole data while the operators and cashiers can access data as per their tasks assigned.

- In the banks like SBI and ICICI where the number of customers or account holders is more, it is not sifficient to keep only one cashier to reciept cash and one cashier to pay cash. For this tasks there are multiple cashiers.
- Consider, account holder 'A' wants to withdraw some amount from his bank account. In this case as shown in Fig. 5.1.4, there are number of cashiers and 'A' can goto any one of them. This is possible because the data of all the account holders is available on the server. And all the cashiers can access the data. This definitely increases the efficieny of bank.

Advantages of Client Server Architecture

- Organizations always try to maintain service and quality competition to sustain its market position with the help of advanced technology where the client/server model makes an effective impact.
- Implementation of client/server architecture in an organization will definitely increase productivity through the usage of cost-effective user interfaces, enhanced data storage, strong connectivity and reliable application services.
- There are number of advantages of Client Server Architecture

1. **Centralized Database :** The database is centrly available for all the clients easily. The centralized database is easy to manage for database administrator.
2. **Security :** Rules defining security and access rights can be defined at the time of set-up of server.
3. **Back-up and Recovery possible :** As all the data is stored on server, it is easy to take periodical backup of it. Also, in case of any break-down if data is lost, it can be recovered easily and efficiently.
4. **Upgradation and Scalability :** Making Changes can is easy by just upgrading the server. The new resources and systems can be added by making required changes in server.
5. **Server Role :** Server can play different roles for different clients.

5.1.3.1 Types of Client Server Database Architecture

Now to understand types of Client Server database architecture first we have to study the concept of layers or services.

Layers or Services

A software application is created using programming languages(called as frontend) and database(called as backend). In every software we have to implement following three layers.

1. **User Layer(presentation layer)**
   - It is also called as client layer which contain User interface of our application. This layer is used for design purpose. In this data is presented to the user and also input can accepted from the user.
   - For example in banking software, the registration form of an account holder can be considerd as user layer.
2. **Business Layer**
   - This layer is also known as business layer. In this layer we can write all business logic like validation of data, calculations, data insertion etc. This acts as an interface between Client layer and Data Access Layer.
   - This layer is also called the intermediary layer helps to make communication faster between client and data layer.
3. **Data Layer**
   - In this layer actual database is comes in the picture. Data Access Layer contains methods to connect with database and to perform insert, update, delete, get data from database based on our input data.
   - Depending upon the implementation of these three layers there are types of database architecture.

A. Two tier architecture
B. Three tier architecture

A. Two Tier Architecture

SPPU - Dec. 13

University Question

Q. Explain two-tier architecture. (Dec. 2013, 2 Marks)

- The **two tier architecture** is based on the client server architecture. The direct communication takes place between client and server.
- The two tier architecture is the architecture in which user interface is run on client side and data layer is stored on the server side. In two tier architecture we can integrate business layer with either presentation layer or database layer or can be distributed in both.

i) The business layer can be integrated with presentation layer at client side. In this case the size of client application increases, hence it is known as **Fat Client.**

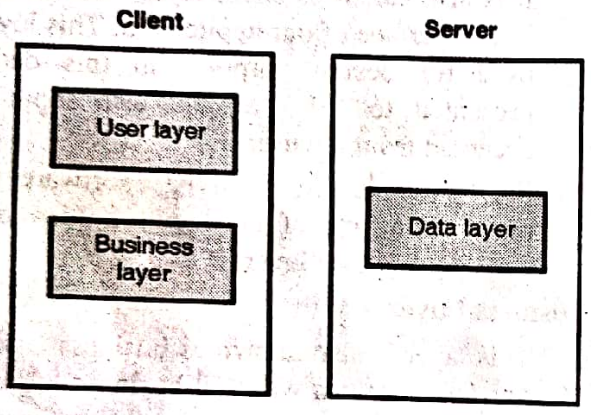


Fig. 5.1.5 : Fat client

ii) The business layer can be integrated with data layer at server side. In this case the size of server application increases, hence it is known as **Fat Server.**

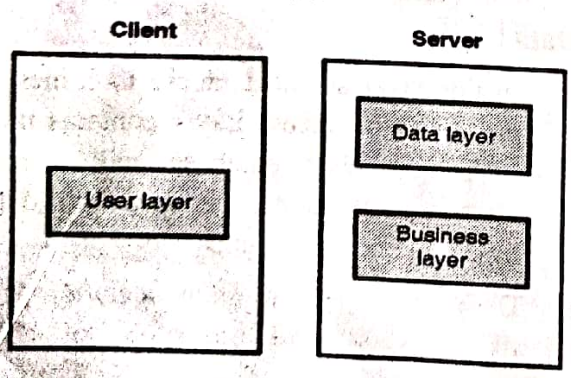


Fig. 5.1.6 : Fat server

iii) The business layer can be integrated with both user layer and data layer.

Advantages of two tier architecture

1. In two tier architecture, applications can be easily developed due to simplicity.
2. In this client and server are directly connected, due to which communication becomes faster.
3. Maximum user satisfaction is achieved with accurate and fast prototyping of applications through robust tools.
4. It contains static business rules which are easily applicable for homogeneous environment.
5. We can integrate application layer physically with the database layer as well as user interface layer

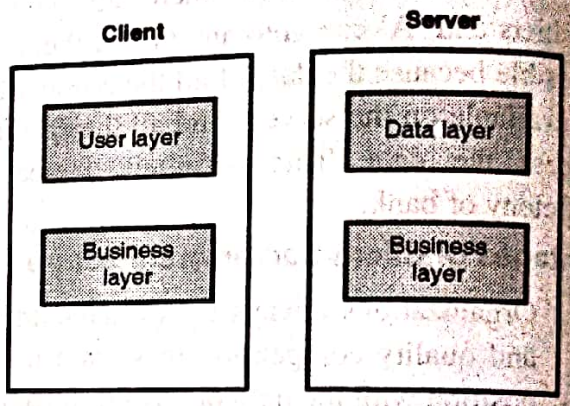


Fig. 5.1.7 : Two tier architecture

Disadvantages of two tier Architecture

1. It can only support to the limited number of users due to lack of scalability.
2. The performance of two tier architecture degrades when number of user increases.
3. Two tier architecture is cost ineffective.
4. As per security concern it is complicated.

B. Three Tier Architecture

SPPU - Dec. 13

University Question

Q. Explain three-tier architecture.

(Dec. 2013, 2 Marks)

- **The three tier architecture** is most widely used architecture in today's world.
- In this architecture the user layer, business layer and data layer are implemented independently by three different applications.
- The data required by the business logic exists in database server.
- In three tier architecture all layers interact with each other independently.

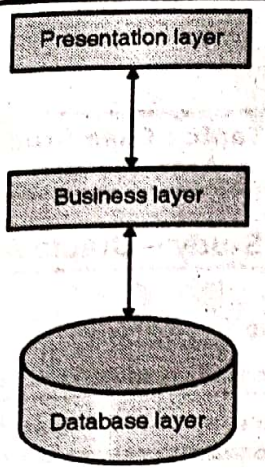


Fig. 5.1.8 : Tree tier architecture

Advantages of Three tier Architecture

1. In three tier architecture we can manage the data independently.
2. We can make the changes in presentation layer without affecting other two tiers.
3. As each tier is independent it is possible to use different groups of developers
4. It is most secure since the client doesn't have direct access to the database layer.
5. When one tier fails there is no data loss, because you are always secure by accessing the other tier.
6. Due to distributed deployment of application server, scalability is increased.
7. A similar logic can be used in various applications. It is reusable.
8. It is robust and secure due to multiple layers.

Disadvantages of three tier architecture

1. It is more complex structure.
2. More difficult to set up and maintain it.
3. The physical separation of the tiers may affect the performance.

Example : SPPU - May 15, May 16

University Question

Q. Explain 3-tier web architecture with diagram for online shopping database system? **(May 2015, May 2016, 8 Marks)**

This is the online shopping diagram for 3 Tier architecture. Here the as a frontend Dot Net environment is used while as backend database MS SQL Server 2008 is used.

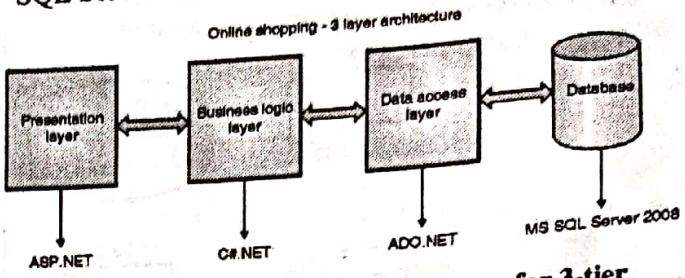


Fig. 5.1.9 : Online shopping diagram for 3-tier architecture

Fig. 5.1.9 shows three layers :

1. **Presentation layer :** This layer consist of user interface designed for the interaction with end user. This layer is created in ASP .Net. It includes the screens which will be used by the end user for shopping. Theses screen show the products with details as per their categories. User can select the product to purchase and add them into cart. This design is created with advanced controls available in ASP .Net.
2. **Business Layer** : This layer consist of validation checking code related to product selection of user. Accidently user may select wrong number of products to purchase. For example, if any user is giving order to purchase 10000 TV sets, then the order should be validate. This logical code is implemented using the C# .Net.
3. **Data Layer** : This layer contains the code interacting with database on the server. For example, accessing product details from database, inserting transaction details of user order in database etc. This database handling is implemented using the ADO .Net. Here all the three layers work independently and efficiently.

5.2 Centralized Database Architecture

SPPU - Dec. 13, May 14

University Questions

Q. Write short note on centralized database system. **(Dec. 2013, 3 Marks)**

Q. Explain centralized database architecture. **(May 2014, 3 Marks)**

A database which is located, stored and

maintained at single location is known as **centralized database architecture**. This location is most often a central computer or database system, for example a desktop or server CPU, or a mainframe computer.

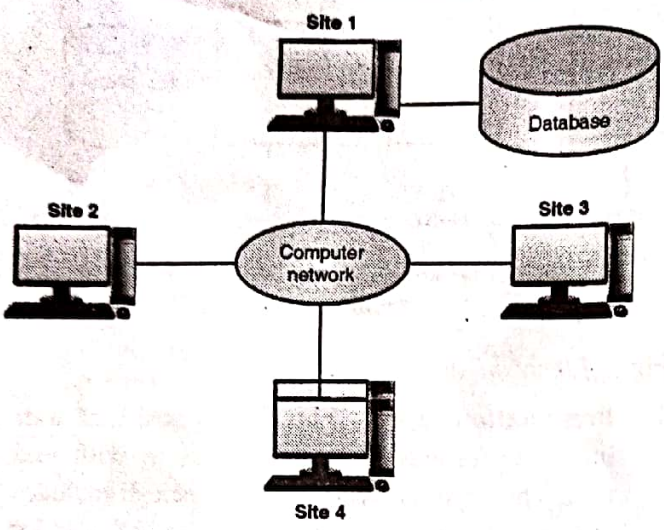


Fig. 5.2.1 : Centralized database architecture

In centralized database architecture the database physically resides on only one site and other sites can access it through the network.

Advantages of centralized database system

1. Easy to use by end-users due to its simplicity provided by storing all the data at one place. It helps to maximize Data integrity and also minimized data redundancy.
2. Data reliability is enhanced and accuracy and consistency of data is maintained.
3. It provides better security for stored data. As all the data of any organization is stored on one place, the organization can easily focus on security detail of one place rather than of multiple locations.
4. In a centralized system, information can be changed or updated easily.
5. Centralized system provides fault tolerance facility so the data is preserved in better way.
6. Also it is cost effective approach as this system requires minimum maintenance cost.

Disadvantages of centralized database system

1. It is highly dependent on network connectivity, hence if the network is slower, then the time require to access data from database is also increases.
2. It decreases the efficiency of the system, as there is only one copy of data. If more than one users want to access it then it is not possible.

Syllabus Topic : Case Study – Oracle Architecture

5.3 Case Study – Oracle Architecture

Oracle Database

Oracle database is made up of two components :

1. Instance
2. Database

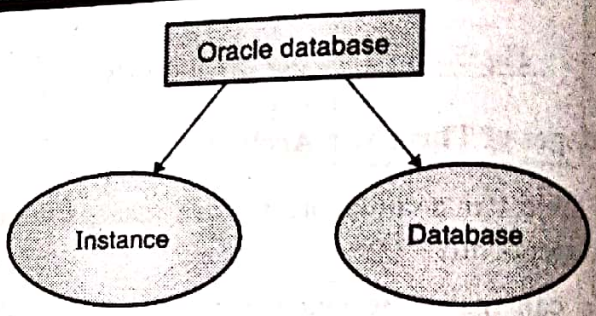


Fig. 5.3.1 : Components of oracle database

1. Instance

In database files the database structure and processes are very important. An instance is nothing but the memory structure and processes that are used to access the data from the database.

The **memory structure** is consist of

- System Global Area (SGA)
- Program Global Area (PGA)
- Optional Software Code.

The background processes are

- Database Writer (DBWn)
- Log Writer (LGWR)
- Checkpoint (CKPT)
- System Monitor(SMON)
- Process Monitor(PMON)
- Optional – Archiver (ARCn), Recoverer (RECO)

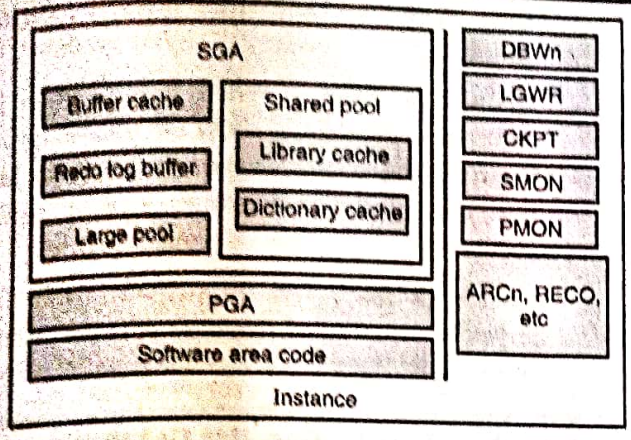


Fig. 5.3.2

Now we will discuss all these components in detail :

- **System Global Area (SGA)**

  This is the primary part of oracle structure. The SGA is a memory area for structures that are shared among users. The main components of SGA are Buffer Cache, Shared Pool, Redo Log Buffer, Large Pool and Java Pool.

  Buffer Cache

  This cache is used to store the frequently accessed data blocks from tables and indices in memory. It reduces the need to perform physical disc IO. The size of this buffer cache can be changes as per the requirement.

  Shared Pool

  - Oracle is designed for multiuser systems. When multiple users executes sme SQL query then they can share the data structures that represent the execution plan for these statements. For this purpose the data the data which is local to each specific call of the statement should be kept in private memory.
  - The Shared pool is used to store the sharable parts of data structures representing the SQL statement with text of the statement. This helps in reducing the compilation time since new call of a statement which is already cached does not have to go through the complete compilation process.
  - The **library cache** is used to store the information about the commonly used SQL statements while the **dictionary cache** is used to store the information about object definition like table, columns, indexes, privileges etc.
  - **Redo Log Buffer :** The DML statements like select, insert, update or delete generates redo entry. This redo entry is nothing but all the information about changes made by user. To store this redo entry, redo log buffer is used before it is written in to redo log files.
  - **Large Pool :** This is basically optional area in the SGA. It is used for I/O processes and it also helps to relieve the burden of shared pool.
  - **Program Global Area (PGA) :** When the SQL statement is parsed, its result is stored in library cache. The value of binding variable is stored in the PGA to make it private so that it should not be accessed by other users.
  - **Software Area Code :** In software area code, the oracle software application resides.

- **Dedicated Server : Process Structure**

  To execute Oracle server code, there are two types of processes : **Server processes** which processes the SQL statements and **background processes** that performs various performance and administrative related tasks. Some of these processes are as follows.

  - **Database Write(DBWn) :** Before removing from the buffer cache, the Database Writer writes the buffer into the disc if it has been modified. It improves the performance of the system.
  - **Log Writer (LGWR) :** The log Writer writes the redo entries from redo log buffer into the redo log files.
  - **Checkpoint (CKPT) :** The checkpoint process updates the headers of the data file when a checkpoint occurs.
  - **System Monitor (CMON) :** This process is used for crash recovery when any process fails. It also manages the space by reclaiming the unutilized space in temporary segments.
  - **Process Monitor(PMON) :** This process is used for process recovery when any process fails. It releases the resources and performs various cleaning operations.
  - **Archiver (ARCn) :** When the online log files fills up, the Archiver copies the online redo log file to an archived redo log.

- **Recoverer (RECO) :** This process resolves failures and conducts cleanups for distributed transactions.

2. Database

The database refers to disc resources. It is basically divided into two types : Logical and Physical structure.

A) Logical Structure : To manage the data efficiently, the oracle database is divided into smaller units like tablespace, segment, extent and data block.

- **TableSpace :** Tablespace is the collection of logical database objects.

  There are three types of tablespaces

  (a) Permanent (b) Undo

  (c) Temporary

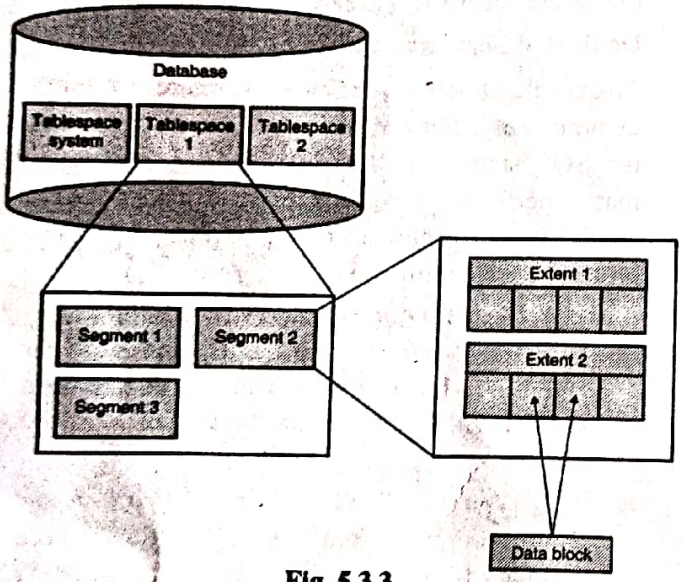


Fig. 5.3.3

- **Segment :** The tablespace is divided into segments. Same type of objects are stored in the segments. There are following types of segments in oracle :

  Table, Index, Cluster, Rollback, Temporary, Cache etc.

- **Extent :** A segment is then divided into extents. It consists of data blocks. An extent is allocated for the enlarged database object.

- **Data Block :** It is the smallest unit of storage in the oracle database.

B) Physical Structure : It consists of data files, redo log files and control files.

- **Data Files :** Data files corresponds to tablespace

- **Redo Log Files :** When a transaction is committed the details about the transaction in the redo log buffer is written into redo log file. This file helps in recovery when failure occurs.

- **Control Files :** The information about the physical structure of database is stored in the control file.

Syllabus Topic : Parallel Database

5.4 Parallel Database

Now a day organizations need to handle huge amount of data with high transfer rate. For such requirement the client server or centralized system is not efficient. The need to improve the efficiency of system, the concept of Parallel Databases comes in picture.

Parallel database improves the performance of processing of data using multiple resources simultaneously. Multiple CPU, Disks can be used simultaneously. A parallel database improves speed of data processing. A parallel server can allow access to a single database by users on multiple machines, with increased performance. By doing parallelization of loading data, building indexes and evaluating queries, the parallel database system improves the performance. In parallel database system we can use thousands of small processors.

Advantages of Parallel Database

1. **Performance Improvement :** By connecting multiple resources like CPU and disks in parallel we can significantly improve the performance of system.

2. **High Availability :** Same data can be stored on multiple locations so that the availability of data can be increased. In parallel database nodes has less contact with each other, so failure at one node does not cause for failure of entire system. One of the surviving nodes recovers the failed node and the system continues to provide data access to users. This means data is much more available than it would be with a single node upon node failure. This also amounts to significantly [illegible] database availability.

3. **It increases Reliability :** When a site fails, the execution can continue with another available site which is having copy of the data. Due to this it becomes more reliable.

4. **It have large capacity :** In parallel database more users request access to the database due to which administrator add more computers to the parallel server. The addition of computers boosts the overall capacity.

Syllabus Topic : Speedup and Scaleup

5.4.1 Speedup and Scaleup

SPPU - May 15, May 16, Dec. 16

University Questions

Q. Explain speedup and scaleup in parallel databases in detail.
(May 2015, May 2016, 5 Marks)

Q. What is speedup and scaleup attributes in parallel database architectures?
(Dec. 2016, 4 Marks)

To measure the performance of parallel processing we can use two important properties:

1. Speedup

- The extent in which more hardware can perform the same task in less time than the original system is known as **Speedup.** It means the execution of task done in less time by increasing degree of parallelism. The time which is required to process a task is inversely proportional to the time when number of resources are used.
- With good speedup, additional processors reduce system response time. We can measure speedup is as follows:

$$\text{Speedup} = \frac{\text{time\_original}}{\text{time\_parallel}}$$

Where,

Time_Parallel is the time spent by a larger, parallel system on the given task

- For example, If the original system took 60 seconds to perform a task, and two parallel systems took 30 seconds, then the value of speedup would be 60/30 = 2.

Linear Speedup

We can say that speedup is linear when speedup is equal to N. That is the elapsed time of small system is N times larger than the elapsed time of large system where N is the number of resources.

Sub-linear Speedup

When speedup is less than N then it is called as sub-linear speedup.

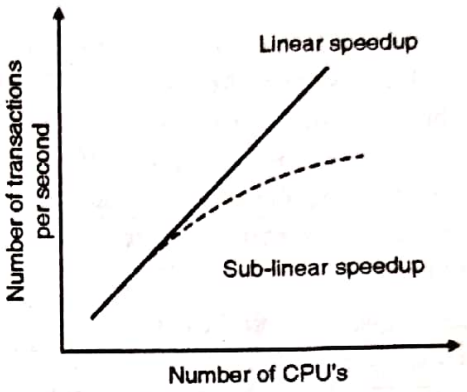


Fig. 5.4.1 : Speedup

2. Scaleup

- The ability to keep the same performance level when both workload and resources increase proportionally is known as **Scaleup.** It is the process of handling large task in same amount of time by increasing degree of parallelism.

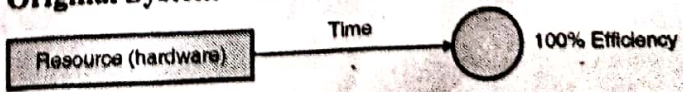
We can measure the ScaleUp is as follows:

$$\text{Speedup} = \frac{\text{Volume\_parallel}}{\text{Volume\_original}}$$

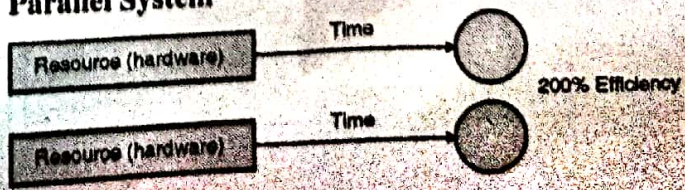
Where Volume_Parallel is the transaction volume processed in a given amount of time on parallel system.

- In case of good Scaleup if there is increase in transaction volume, then to keep response time constant, we can add hardware resources like CPU.

Original System



Parallel System



5.4.1.1 Different Factors Affecting the Speedup and Scaleup Attributes

SPPU - Dec. 16

University Question

Q. Explain the different factors affecting the speedup and scale-up attributes. **(Dec. 2016, 4 Marks)**

Startup costs

For all the parallel operations, there is a associated cost involved in starting the process. When a big process is break in small processes then it consumes some amount of time and recourses. If we want to execute a query in parallel then we need to partition the table and instructs various processors to execute the query in parallel.

Consider a query to sort the data.

```
Select * from emp order by ename
```

Now to execute this query in parallel we need to partition the table as per sorting criteria and instructs various processors to execute the query in parallel. Both of these operations need some amount of time before the parallel execution.

- **Interference :** The various resources are shared by all he processes which executes in parallel. Whenever interference of a new process happens, it leads to slow down the overall access of all the processes. This affects both speed-up and scale-up.
- **Skew :** It is difficult to break a big tasks in equal size small tasks. In such case the performance of the system depends upon the slowest CPU which processes the largest sub task. This type of uneven division of big tasks into smaller ones is called as skew. This also affects the speed-up and scale-up.

Syllabus Topic : Architecture of Parallel Databases

5.4.2 Architecture of Parallel Database

SPPU - Dec. 14, May 16

University Questions

Q. Explain any two parallel database architectures in details. **(Dec. 2014, 5 Marks)**

Q. Explain parallel database architectures. **(May 2016, 4 Marks)**

There are four different architectural models of parallel database:

1. Shared memory	2. Shared disk
3. Shared nothing	4. Hierarchical

5.4.2.1 Shared Memory

SPPU - Dec. 15

University Question

Q. Explain Shared Memory Parallel Database System architecture. **(Dec. 2015, 3 Marks)**

In this architecture of parallel database common memory is shared among the multiple processors. These processors are connected through the interconnection network to the main memory and disk. The connection used in this architecture is usually high speed network connection which makes data sharing easy. Shared memory architecture have large amount of cache memory at each processors.

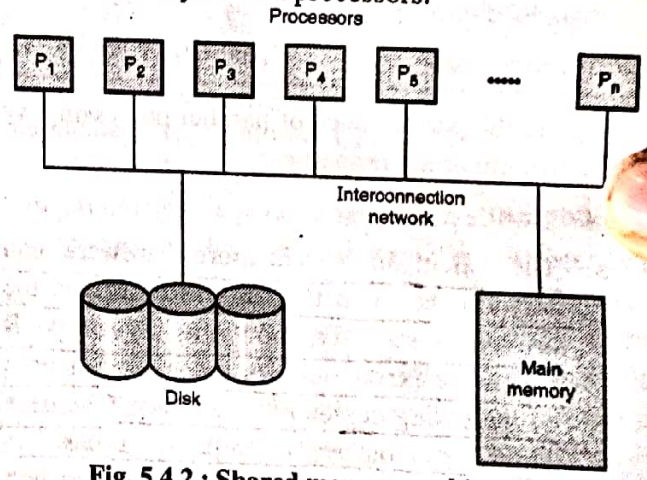


Fig. 5.4.2 : Shared memory architecture

Advantages of shared memory architecture

1. Any processor can access data easily.
2. Effective communication between processors through common memory address space.
3. Communication overheads are less.

Disadvantages of shared memory architecture

1. Waiting time of processors is increased due to more number of processors
2. Degree of parallelism is limited.
3. Addition of processor slows down the existing processors.
4. When a processor tries to access the data updated by other processors, then we have to take care that the data is of latest updated version.

5.4.2.2 Shared Disk Architecture

In the shared disk architecture of parallel database system single disk is shared among all the processors. These all processors also have their own private memory which makes data sharing efficient.

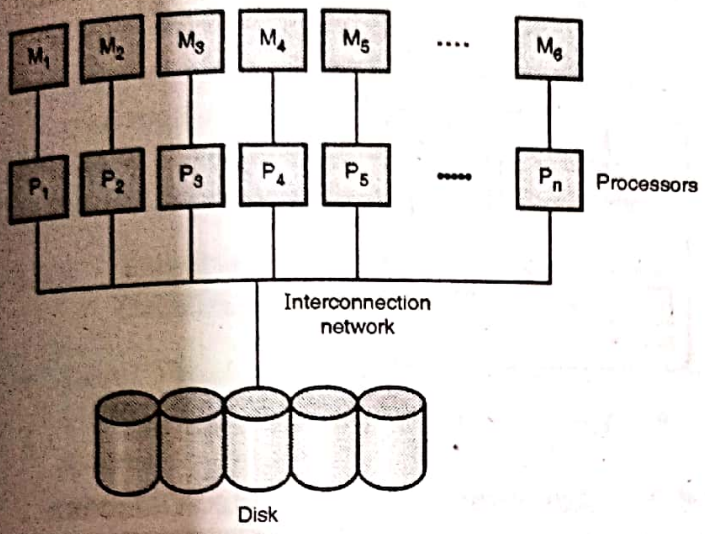


Fig. 5.4.3 : Shared disk architecture

Advantages of shared disk architecture

1. It has fault tolerance means failure of any processor does not lead to execution stop; rather any other processor completes the task.
2. As compared to shared memory architecture it supports large number of processors.
3. This architecture permits high availability.
4. Interconnection to the memory is more smooth and free.

Disadvantages of shared disk architecture

1. The scalability of Shared disc system is limited as large amount of data travels through the interconnection channel.
2. The speed of existing processors may slow down if more processors get added.

5.4.2.3 Shared Nothing Architecture

SPPU - Dec. 15

University Question

Q. Explain Shared Nothing Parallel Database System architecture. **(Dec. 2015, 7 Marks)**

- The shared nothing architecture is a distributed computing architecture in which each node is independent. More specifically, none of the nodes share memory or disk storage.
- In this architecture each processor has its own local memory and local disk.
- Intercommunication channel is used by the processors to communicate.
- The processors can independently act as a server to serve the data of local disk.

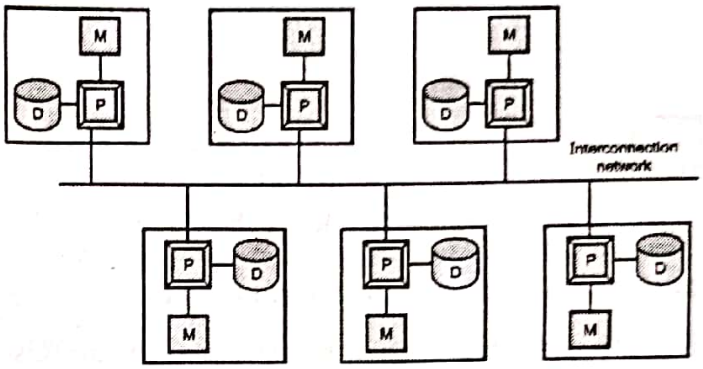


Fig. 5.4.4 : Shared nothing architecture

Advantages of shared nothing architecture

1. This architecture is scalable regarding the number of processors. Increase in their number is easy and flexible.
2. When nodes get added transmission capacity increases.
3. It is a read only database and decision support application.
4. In this architecture failure is local. It means that failure of one node can not affect to other nodes.

Disadvantages of shared nothing architecture

1. The cost of communication is higher than shared memory and shared disk architecture.
2. The data sending involves the software interaction.
3. In this technique more coordination is required.

5.4.2.4 Hierarchical Architecture

Hierarchical model architecture is also known as Non Uniform Memory Architecture. This is nothing but the combination of shared disk, shared memory and shared nothing architecture.

Consider there are two groups of processors say A and B. All processors from both the groups have local

memory. But processors from other groups can access memory which is associated with the other group in coherent.

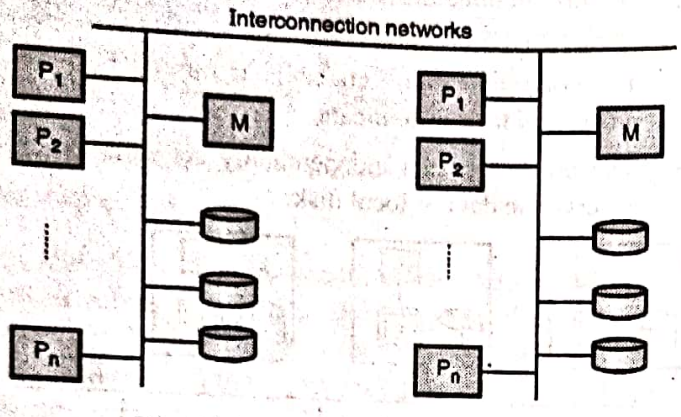


Fig. 5.4.5 : Hierarchical architecture

Advantages of Hierarchical Architecture

1. The availability of memory is more in this architecture.
2. The scalability of system is also more.

Disadvantages of Hierarchical Architecture

1. This architecture is costly as compared to other architectures.

Syllabus Topic : Distributed Database

5.5 Distributed Database

SPPU - Dec. 13, Dec. 15, May 16

University Questions

Q. Write a short note on : Distributed Database System. **(Dec. 2013, 3 Marks)**

Q. Define Distributed Database. **(Dec. 2015, May 2016, 2 Marks)**

5.5.1 Distributed Database Introduction

Distributed database is the type of database management system in which number of databases are stored at various locations and interconnected through the computer networks. Means we can say that, "Distributed database is a collection of various interconnected databases which are physically spread at various locations and communicate via a computer network. In distributed database system each site may have its own memory and its own Database Server.

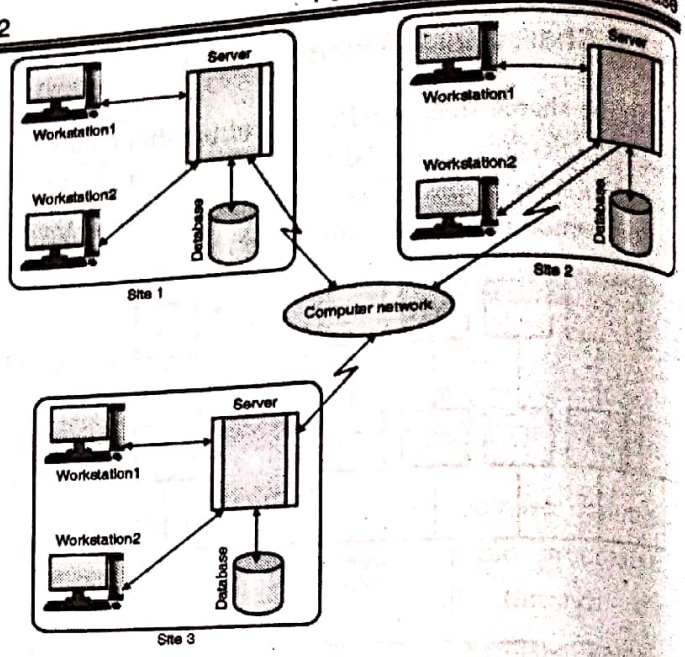


Fig. 5.5.1 : Distributed Database System

5.5.2 Advantages of Distributed Database

SPPU - May 14, Dec. 15

University Question

Q. What are advantages and disadvantages of Distributed Database system architecture? **(May 2014, Dec. 2015, 6 Marks)**

1. **Modular Development :** If we need to expand the same system at new locations, we can use distributed databases, we simply requires to add new computers and local data to the new site and have to connect them to the communication network. It does not interrupt the current functionality.
2. **Increases reliability :** The possibility of the system running at any point is known as reliability. In distributed database system if one component fails to work, other component can take over its function. Due to this the whole system does not affected.
3. **Improve performance :** In distributed database system, large database is distributed among the multiple sites. Due to this distribution a small database is exist at each site. The small database is easy to handle which increases the performance.
4. **Increase availability :** In distributed database system data is scattered at various nodes. Therefore if one node fails then data can be easily available from another node. It means that it have

higher availability.

5. **Faster response :** In this system most of the data is local. Due to this user request can be answered quickly.
6. **Local control :** The sites which contain the data are the owner of the data. These sites can locally control the data.
7. Even though the data is distributed, the system presents the data to the user as if it were located locally. User doesn't know where the data is located physically.

5.5.3 Disadvantages of Distributed Database

C S D I C

1. **Cost :** The requirement of additional hardware to establish a network between sites causes increment of cost. There are ongoing communication costs incurred with the use of this network. There are also additional manpower costs to manage and maintain the local DBMSs and the underlying network.
2. **Security :** In distributed database system database has access from all the sites therefore security becomes an issue for the distributed database system.
3. **Database design more complex :** In distributed database system we have to distribute the large database on various sites in the form of small databases. Due to this, designing of database become more complex.
4. **Increased processing overhead :** It required many messages to share within the sites to complete a distributed transaction.
5. **Complex data integrity :** In distributed database data integrity becomes complex. It requires too many resources to integrate the data.

5.5.4 Types of Distributed Database System

SPPU - Dec. 14, May 16

University Questions

Q. Compare homogeneous and heterogeneous distributed database. **(Dec. 2014, 5 Marks)**

Q. Explain homogeneous and heterogeneous Distributed Databases? **(May 2016, 6 Marks)**

There are two different types of distributed database.

1. Homogenous Distributed Database Systems
2. Heterogeneous Distributed Database Systems

5.5.4.1 Homogeneous Distributed Database Systems

In homogeneous DDBMS, all sites use the same database management system product. In homogeneous distributed database, as all sites have identical software, they are aware of each other and agree to cooperate in processing user requests. A homogeneous DBMS appears to the user as a single system.

It is much easier to design and manage. This design provides incremental growth by making additional new sites to DDBMS easily. It allows increased performance by exploiting the parallel processing capability of multiple sites.

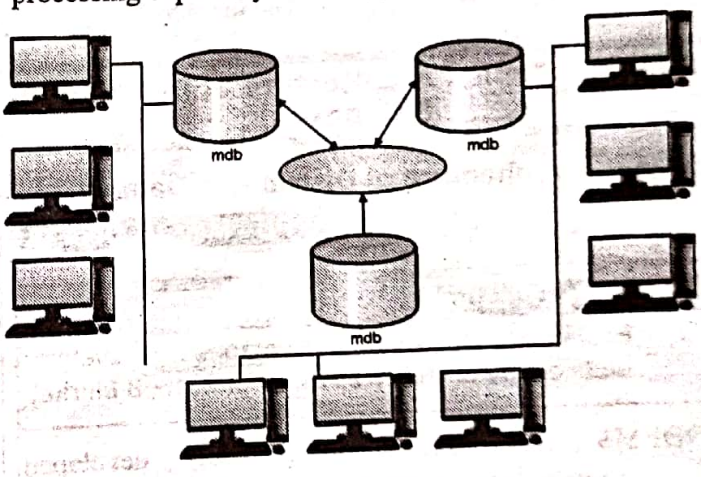


Fig. 5.5.2 : Homogeneous DDBMS

The following conditions must be satisfied for homogeneous database:

- The operating systems used at each location must be same or compatible.
- The database applications used at each location must be same or compatible
- The data structures used at each location must be same or compatible.

5.5.4.2 Heterogeneous Distributed Database System

The word heterogeneous state that the ability to

accept different forms of databases. In heterogeneous database we can use different types of databases. In this type we can use different schemas. In a heterogeneous system, sites may run different DBMS products, which need not based on the same underlying data model, and so the system may be composed of relational, networked, hierarchical and object-oriented DBMSs. In this system, different computers and different operating systems or data models can be used at the each of the location.

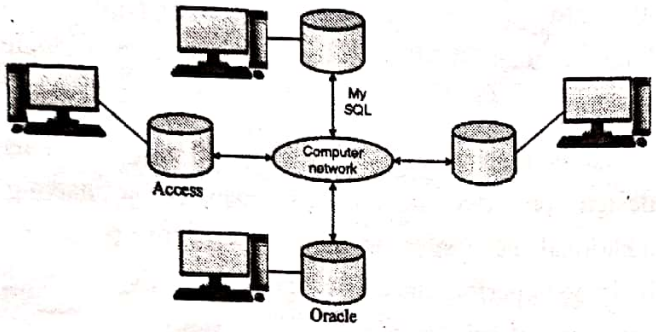


Fig. 5.5.3 : Heterogeneous Database System

Syllabus Topic : Architecture of Distributed Databases

5.5.5 Architecture of Distributed Database

SPPU - May 16, Dec. 16

University Question

Q. Explain distributed database system architecture. (May 2016, Dec. 2016, 8 Marks)

DDBMS architectures are generally developed depending on three parameters:

- **Distribution :** It states the physical distribution of data across the different sites.
- **Autonomy :** It indicates the distribution control of the database system and the degree to which each constituent DBMS can operate independently.
- **Heterogeneity :** It refers to the uniformity or dissimilarity of the data models, system components and databases.

Architectural Models

Some of the common architectural models are :

1. Client-Server Architecture for DDBMS
2. Peer-to-Peer Architecture for DDBMS
3. Multi-DBMS Architecture

5.5.5.1 Client-Server Architecture for DDBMS

Client-server architecture is a two-level architecture where the functionality of architecture is divided into servers and clients. Data management, query processing, optimization and transaction management are the functions of server. Client functions include mainly user interface, consistency checking and transaction management.

The two different client-server architectures are :

1. Single Server Multiple Client
2. Multiple Server Multiple Client

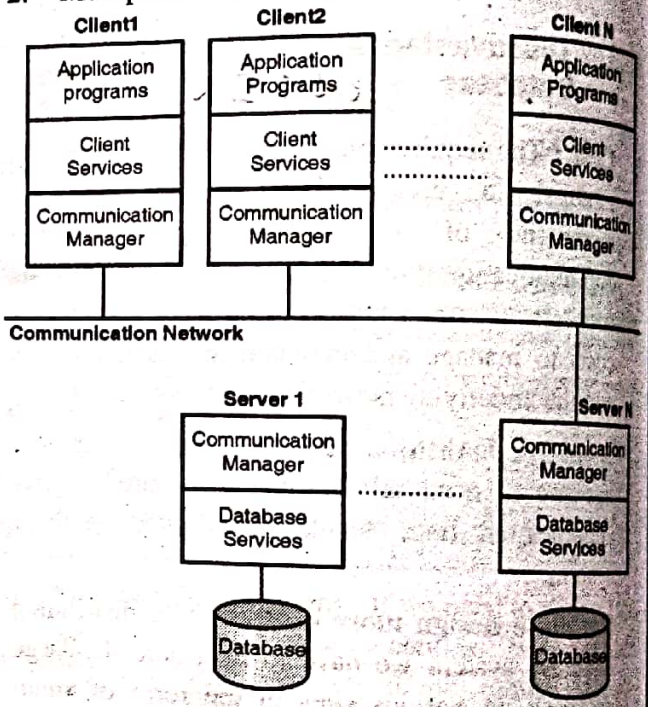


Fig. 5.5.4 : Client-server architecture

5.5.5.2 Peer-to-Peer Architecture for DDBMS

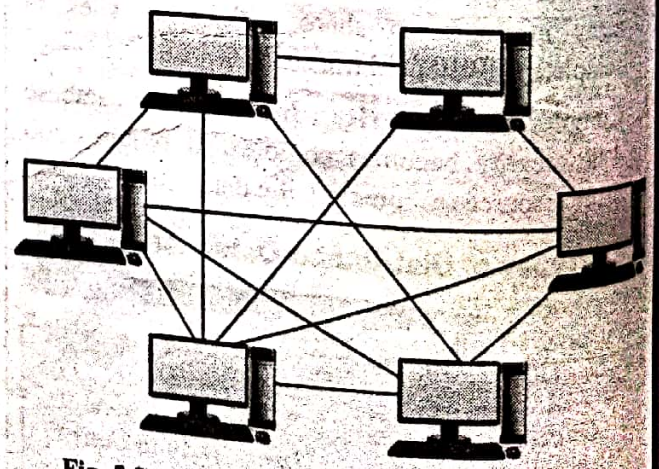


Fig. 5.5.5 : Peer-to-peer architecture for DDBMS

- The network architecture in which each workstation or node has same capabilities and

responsibilities is known as peer-to-peer architecture. Peer-to-peer may also be used to refer single software design. Due to this each peer may act as both client and server.

- The peers share their resources with other peers and co-ordinate their activities.

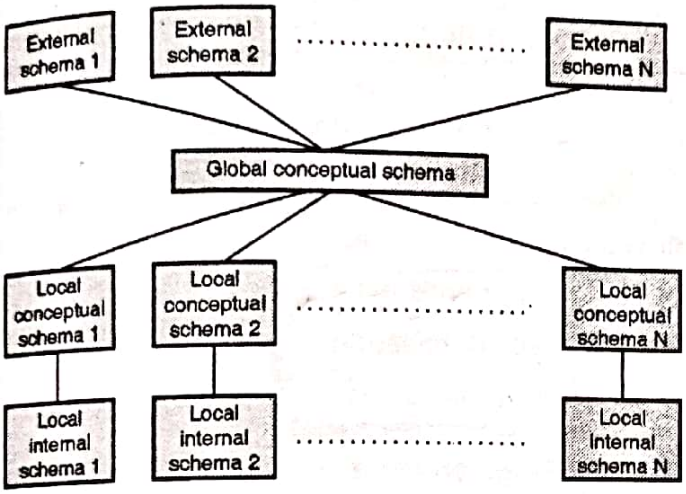


Fig. 5.5.6

- This architecture generally has four levels of schemas
  1. **Global Conceptual Schema :** As the name suggest, it shows only the global logical view of data.
  2. **Local Conceptual Schema :** It only shows logical data organization at each site. The data on every site is local for that particular site.
  3. **Local Internal Schema :** Local internal schema shows the physical data organization at each and every site.
  4. **External Schema :** External schema shows the external user view.

5.5.5.3 Multi - DBMS Architectures

Multi-database management system architecture is an integrated database system formed by collection of two or more autonomous database systems. Multi-DBMS can be expressed through six levels of schemas.

- **Multi-database View Level :** It shows multiple user views which consists of subsets of the integrated distributed database.
- **Multi-database Conceptual Level :** It shows the integrated multi-database that consists of global logical multi-database structure definitions.
- **Multi-database Internal Level :** It shows the data distribution across different sites and multi-database to local data mapping.
- **Local database View Level :** It shows the public view of local data.
- **Local database Conceptual Level :** It shows the local data organization at each site.
- **Local database Internal Level :** It shows the physical data organization at each site.

There are two design alternatives for multi-DBMS

- Model with multi-database conceptual level.

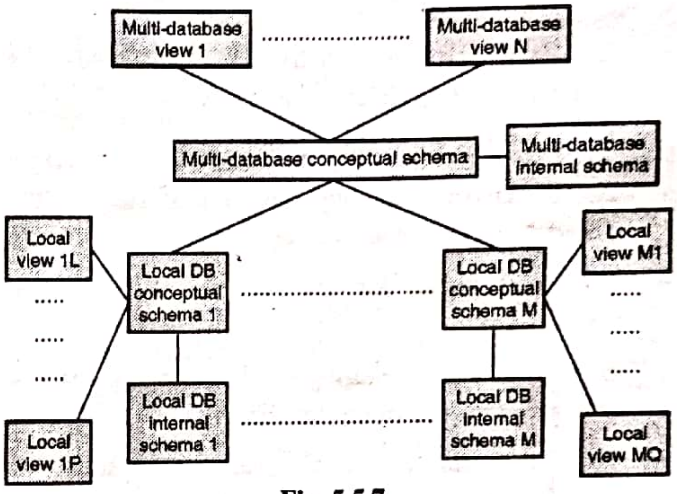


Fig. 5.5.7

- Model without multi-database conceptual level.

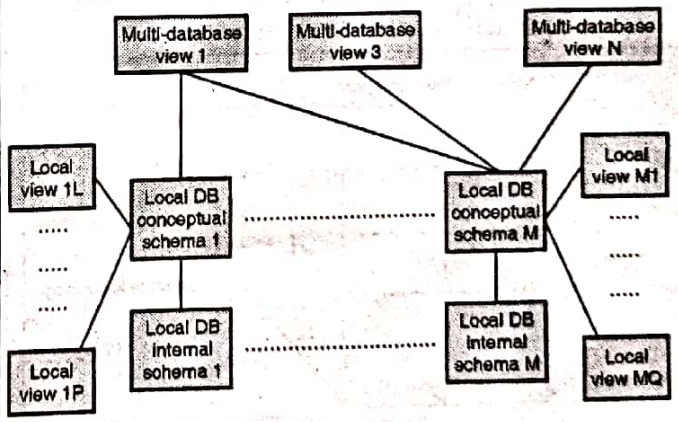


Fig. 5.5.8

Syllabus Topic : Distributed Database Design

5.5.6 Distributed Database Design

The design of distributed database includes

A) Design problem
B) Design strategies(top-down, bottom-up)
C) Distributed Database Storage
- Fragmentation
- Data replication
- Data Allocation

5.5.6.1 Design Problem

In **designing a distributed database**, you must decide which portion of the database and programs are to be stored at which location. It also includes the designing of network itself.

- In designing of distributed database the distribution of application involves
  1. DDBMS software distribution
  2. Distribution of application that run on the database
- Important points for the analysis of distributed systems are -
  - The level of sharing - data sharing, data + program sharing or no sharing
  - Behaviour of access patterns - static or dynamic
  - Level of knowledge on access pattern behavior - partial information, complete information or no information.

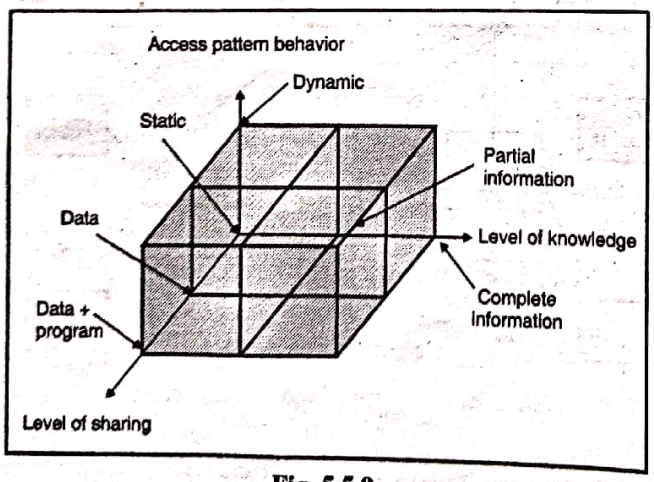


Fig. 5.5.9

5.5.6.2 Design Strategies

Top-down approach

- Designing systems from initial stage
- Homogeneous systems

Bottom-up approach

- The databases considered to be exist at a number of sites already
- To solve common tasks the databases should be connected

Syllabus Topic : Distributed Data Storage

5.5.6.3 Distributed Data Storage

To store a relation in distributed database design there are following approaches

A) Fragmentation
B) Data Replication
C) Data Allocation

A) Data Fragmentation

The data fragmentation is the technique used in distributed database design. This technique is used to break up the database into logical units called **fragments**. The information of fragmentation is stored into the catalogue of distributed database system which is used by the processing computer to process the user's request. There are three different types of fragmentation :

1. **Horizontal Fragmentation :** The fragmentation of relation is done horizontally, in the form of rows. Each fragment which contains unique rows is stored in different computer nodes. Each horizontal fragment may have a different number of rows, but each fragment must have the same number of attributes.
2. **Vertical Fragmentation :** The division of relation into fragments consist of collection of attributes. In the vertical fragments there can be same number of rows and can have different attributes which depends upon the key.
3. **Mixed Fragmentation :** Mixed Fragmentation is the combination of both horizontal and vertical fragmentation. It is the two-step process. The first step is to achieve the horizontal fragmentation to obtain the necessary rows. And second step is to achieve vertical fragmentation to divide attributes among the rows.

B) Data Replication

The data replication is the storage of copies of data at multiple sites on the network. Fragmented copies can be stored at various sites. The data replication enhances data availability and response time. All copies of fragmented data must be identical. The maintenance of replication may become complex. A database can be either **fully replicated, partially replicated** or **un-replicated.** We can use data replication while we have to handle large database. Due to replication it becomes possible to retrieve the lost data. We can retrieve lost data from the other sites.

1. **Fully replicated :** The fully replicated database stores multiple copies of entire database on all the sites. The availability of entire database on all the sites leads to fast processing of queries.
2. **Partially replicated :** Portion of tables(relations) is stores on different sites. The frequency of data access is considered while distributing the data on different sites.
3. **Non replicated or No replication :** Non replicated replication stores single copy of database fragment at a single site. There is no duplication occurs in this phase.

C) Data Allocation

The data allocation decides the locations of different data for storage purpose. Data allocation can be centralized, partitioned or replicated.

1. **Centralized :** The data can be stored on a single particular site. There is no distribution of database.
2. **Partitioned :** The database get divided into multiple fragments and stored on various sites.
3. **Replicated :** Copies of one or more database fragments are stored at several sites.

Syllabus Topic : Distributed Transaction Basics

5.6 Distributed Transaction

A **distributed transaction** contains one or more statements which make updations in the data of two or more distinct nodes of a distributed database.

5.6.1 Distributed Transaction Basics

There are two types of transaction we can consider :

1. **Local Transaction :** Local transactions are the transactions which perform operation only on the single local database. These transactions can access and update the data from local database only.
2. **Global Transaction :** Global transactions are the transactions which perform operation on the several local databases. It can access and update several local databases.

For distributed transaction, we will study the system structure of distributed database and its possible failure modes. Maintaining ACID properties of transaction is very important task which is done by a utility called as Transaction Manager. All sites have their own Transaction Managers which co-operate each other to execute the transactions successfully.

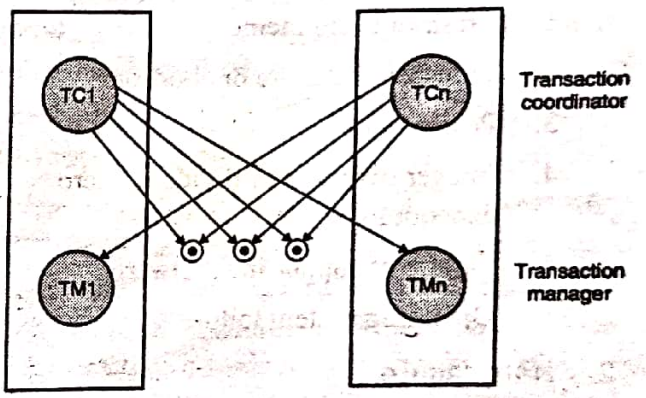


Fig. 5.6.1

Now to understand the working of Transaction Manager we will consider a system with two subsystems.

- **Transaction Manager :** The execution of transactions which access the local data is managed by the Transaction Manager. These transactions may be local or part of any global transaction. The Transaction Manager should participate in appropriate concurrency control scheme to coordinate the concurrent execution of transactions executing that site.
- **Transaction Coordinator :** At the site, the execution of all local and global transactions is coordinated by the Transaction Coordinator. The Transaction coordinator should start the execution of transaction.

Break the transaction in sub-transactions and assign it to different sites. Coordinate the completion of transaction, in which the transaction either committed or aborted.

Syllabus Topic : Failure Modes

5.6.2 Failure Modes

In distributed transactions failure can be broadly categorized into three different groups. They are as follows:

1. Soft Failure

This type of failure can cause the loss of volatile memory of the computer. Due to this we lost the information stored in the non-persistent storage like main memory, buffers, caches or registers. It is also known as system crash. The various types of soft failures are as follows :

1. Crash of Main memory
2. Transaction failure or abortion.
3. Power failure.
4. Integer overflow or divide-by-zero exceptions.
5. Failure of supporting software.
6. Operating system failure.

2. Hard failure

Hard failure causes the loss of data which is stored in non-volatile storage such as Disk. It is also known as Disk Failure. Because of failure in disk, corruption of data in some disk blocks or failure of the total disk may occur. The causes of a hard failure are as follows :

1. Faults in media.
2. Malfunction in Read-write operations
3. Information Corruption on the disk.
4. Power failure.
5. Read/write head crash of disk.

If we have new, formatted and ready-to-use disk in backup then recovery from disk failure can be easy and short.

3. Network Failure

Network failures are widely spread in distributed or network databases. It consist of the errors induced in the database system due to the distributed format of the data in the network and transformation of data.

There are number of reason of network failure

1. Failure in communication link
2. Data corruption in transformation process.
3. Site failures.
4. Congestion in network

Syllabus Topic : Commit Protocols

5.7 Commit Protocols

Atomicity is an important ACID property. To maintain atomicity it is necessary that the final outcome of any transaction should be accepted by all the sites on which it is executing. That means the transaction must be either committed for all the sites or aborted for all the sites. For this purpose the commit protocols are used.

There are different commit protocols. They are as follows:

1. One-phase Commit Protocol (1 PC)
2. Two-phase Commit Protocol (2 PC)
3. Three-phase commit protocol (3 PC)

5.7.1 One-phase Commit Protocol (1 PC)

One phase commit protocol in distributed database is the simplest commit protocol. In distributed transaction there is a controlling site and a number of slave sites where the transaction is being executed. For the One phase commit protocol following steps takes place:

1. When each slave has completed its transaction locally, it sends a done message to the controlling sites.
2. After sending the message the slave sites wait for 'Commit' or 'Abort' message from controlling sites. This waiting time is called as window of vulnerability.

3. When a controlling site receive 'Done' message from slave site, it takes a decision to commit or abort. This is called as Commit Point. Then the controlling site sends this decision to all the slave sites.
4. After receiving this message from controlling sites, the slave sites either commit or abort and then send an acknowledgement message to controlling sites.

5.7.2 Two-phase Commit Protocol (2 PC)

SPPU - Dec. 15

University Question

Q. Explain Two Phase Commit Protocol in Distributed Database (Dec. 2015, 5 Marks)

In distributed database **two phase commit protocols** reduces the waiting time means vulnerability of one phase commit protocol. Two steps are performed in two phase commit protocol as follows:

Step 1: Prepare Phase

- As we have seen in one phase protocol, after completing the transaction locally, each slave sends a 'done' message to the controlling sites. When the controlling sites received 'done' message from all slaves. It sends a prepare message to the slave sites.
- The slaves sites vote on whether they want to commit or not. If a slave wants to commit, it sends a "Ready" message.
- A slave site that does not want to commit, sends a "Not Ready" message to controlling sites. This may happen when the slave site has conflicting concurrent transactions or there is a timeout.

Step 2 : Commit/Abort Phase

- After the controlling site received "Ready" message from all the slave sites :
  - The controlling site sends a "Global Commit" message to the slave sites.
  - The slave sites apply the transaction and send a "Commit ACKNOLEDEGEMET" message to the controlling site.
  - When the controlling site receives "Commit Acknowledgement" message from all the slaves, it considers the transaction as committed.
- After the controlling site has received the first "Not Ready" message from any slave :
  - The controlling site sends a "Global Abort" message to the slaves.
  - The slaves abort the transaction and send "Abort Acknowledgment" message to the controlling site.
  - When the controlling site receives "Abort Acknowledgment" message from all the slaves, it considers the transaction as aborted.

5.7.3 Three-phase Commit Protocol (3 PC)

SPPU - Dec. 15

University Question

Q. How 3 PC is different than 2 PC? (Dec. 2015, 3 Marks)

This protocol contains one more phase '**Prepare to Commit Phase**' than the **2 PC**. We will see the **difference** between these two protocols in the three steps of **3 PC**.

Step 1 : Prepare Phase

- As we have seen in one phase protocol, after completing the transaction locally, each slave sends a 'done' message to the controlling sites. When the controlling sites receives 'done' message from all slaves. It sends a prepare message to the slave sites.
- The slaves sites vote on whether they still want to commit or not. If a slave wants to commit, it sends a "Ready" message.
- A slave site that does not want to commit, it sends a "Not Ready" message to controlling sites. This may happen when the slave site has conflicting concurrent transactions or there is a timeout.

Step 2 : Prepare to Commit Phase

- In this step controlling site issues an "Enter Prepared State" broadcast message.
- The slave sites vote "OK" in response.

Step 3 : Commit / Abort Phase

- After the controlling site receives "Ready" message from all the slave sites–
  - The controlling site sends a "Global Commit" message to the slave sites.

- The slave sites apply the transaction.

– After the controlling site has received the first "Not Ready" message from any slave –

- The controlling site sends a "Global Abort" message to the slaves.
- In this phase commit acknowledgement an abort acknowledgement is not required unlike two phase protocol.

Syllabus Topic : Concurrency Control in Distributed Database

5.8 Concurrency Control in Distributed Database

In distributed database systems, database is typically used by many users. These systems usually allow multiple transactions to run concurrently i.e. at the same time. The activity of coordinating concurrent access to a single database in a multiuser database management system (DBMS) is known as **Concurrency control**. It allows users to access a database in a multi-programmed fashion. In centralized database system, several locking protocols are used for concurrency control. The major difference between centralized and distributed database system is that how lock manager deal with replicated data.

There are some approaches for concurrency control in distributed database :

A) Single Lock Manager

- In distributed database system which have several sites, maintains a single lock manager at a particular chosen site this concept is known as single lock manager approach. Fig. 5.8.1 shows the single lock manager approach.
- Here the sites S1, S2, S3, S4, S5, and S6 are present among those the data D is replicated on S1 S6 and S4. And the site S3 acts as a Lock Manager.
- At the site S2 transaction T1 requests for data item D(step 1). These requests are forwarded by the site S2 to the lock manager's site S3 for granting purpose (step 2).

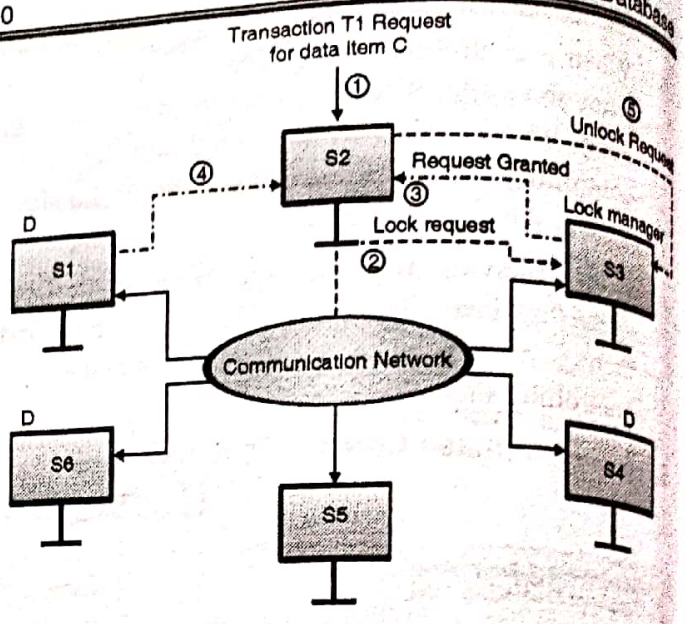


Fig. 5.8.1 : Single lock manager approach

- If the requested data item is free then the request for that data item is granted immediately by the lock manager(step 3).
- If the request is granted then the data item can be read from any site where the replica of required data item is present; here, the site S2 takes data item D from site S1(step 4).
- After the transaction T1 executed successfully the Transaction /manager at site S2 again send the request for unlock the data item so that other transactions can used the data item D now(step 5).

B) Distributed Lock Manager

- In distributed database system there are several sites and data is replicated or fragmented on those sites. The distributed lock manager approach is nothing but the distributing the functionality of lock manager over several sites.
- Fig. 5.8.2 shows the distributed lock manager approach using primary copy protocol.
- Here the sites S1, S2, S3, S4, S5, and S6 are present.
- Data A, B, C is replicated on multiple sites.
- Among the six sites S6 holds primary copy of data item B so the lock manager for granting data access on B is present at only site S6, similarly S5 site acts as a lock manger for data item A and S4 acts as a lock manager for data item C.

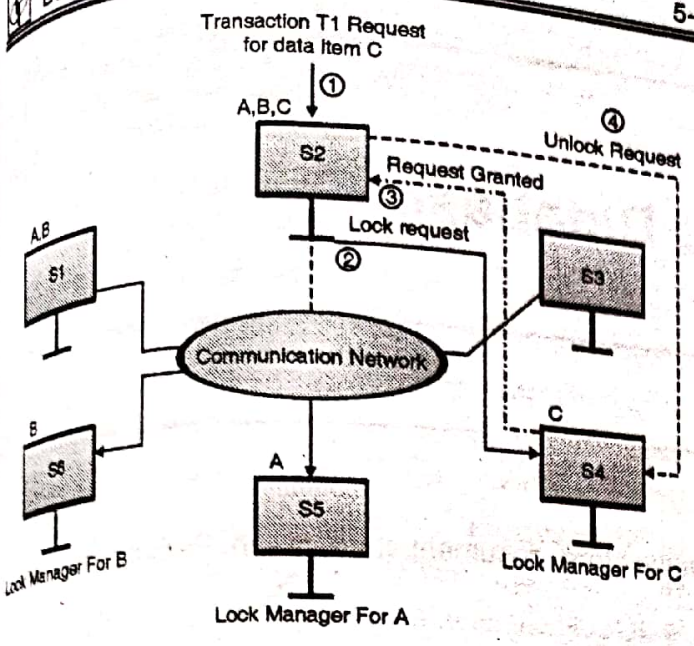


Fig. 5.8.2 : Distributed lock manager approach

- At the site S2 transaction T1 requests for data item C (step 1). Even if the replica of C is present at site S2 locally it is mandatory to first the request lock for accessing C from site S4 as it is acts as lock manager for C. So the transaction manager at site S2 sends lock request to site S4(step 2).
- If the requested data item is free then the request for that data item is granted immediately by the lock manager, hence site S4 grants lock request for C(step 3).
- Now the transaction T1 can access local copy of Data item C present at site S2(step 4).
- After the transaction T1 executed successfully the Transaction manager at site S2 again send the request for unlock the data item so that other transactions can used the data item C now(step 5).

❑❑❑

UNIT VI

CHAPTER 6

NoSQL Database

Syllabus

- Introduction to NoSQL Database
- Types and examples of NoSQL Database- Key value store, document store, graph, Performance
- Structured verses unstructured data
- Distributed Database Model
- CAP theorem and BASE Properties
- Comparative study of SQL and NoSQL
- NoSQL Data Models
- Case Study - unstructured data from social media
- Introduction to Big Data, Hadoop : HDFS, MapReduce

Syllabus Topic : Introduction to NoSQL Database

6.1 Introduction to NoSQL Database

The relational databases are widely used in software industry. The design of relational databases is not such that which can cope with the scale and agility challenges that face modern real time applications, nor were they built to obtain benefit of the commodity storage and processing power available now a day. Now a days the data management becomes very difficult because of the tremendously increase in the size of data in various emerging fields like social networking, e-commerce etc.

- NoSQL is also known as "non SQL" or "non relational" or "Not Only SQL". NoSQL is database which provides a complete different mechanism for storing and retrieval of data which is modelled in means other than the tabular relations used in relational database management systems.
- Sometimes the term NOSQL seems to be confusing as handling data without SQL is out of imagination for some people. But actually the meaning of SQL is Not Only SQL.
- NoSQL challenges the dominance of relational databases. NoSQL databases are increasingly used in big data and real-time web applications.
- In NoSQL the data structures used like key-value, column, graph or document are different from those used in traditional relational database system which makes some operations faster in NoSQL.
- Usually the data structures used by NoSQL databases are also more flexible than relational database system.
- NoSQL allows the insertion of data without a predefined schema (design). In this database, it is possible to make significant application changes in the system without having worry about service interruptions. Because of it, the speed of development increases, the integration of code becomes more reliable and time of database administration decreases.
- To enforce data quality controls, developer has to add application-side code. NoSQL supports validation rules to be applied on the database which helps user to control the data while maintaining the advantage of a dynamic schema.
- NoSQL databases are more concentrated on availability, partition tolerance, and speed for which they may compromise the consistency. The

basic reason of no wide adoption of NoSQL is use of low level query language rather than SQL.

History of NoSQL

- Such database have existed since year 1960, but not known as "NoSQL". Need of this database comes in picture with the rise of web related applications like Google, Facebook, Amazon etc.
- In 1998 Carlo Strozzi first introduced a lightweight, open source relational database system which did not expose the standard Structured Query Language (SQL) interface. Carlo Strozzi gives the name as "NoSQL" to it. He suggested that as the NoSQL is differ from the traditional relational model, it should be called as "NoREAL" means "No Relational".
- Ohan Oskarsson, then a developer at Last.fm, reintroduced the term *NoSQL* in early 2009 when he organized an event to discuss "open source distributed, non relational databases". The name attempted to label the emergence of an increasing number of non-relational, distributed data stores, including open source clones of Google's BigTable/ MapReduce and Amazon's Dynamo.
- Most of the early NoSQL systems did not attempt to provide atomicity, consistency, isolation and durability guarantees, contrary to the prevailing practice among relational database systems.
- Based on 2014 revenue, the NoSQL market leaders are MarkLogic, MongoDB, and Datastax. Based on 2015 popularity rankings, the most popular NoSQL databases are MongoDB, Apache Cassandra, and Redis.

Use NoSQL when needs are like

1. Decentralized applications (e.g. Web and mobile)
2. Continuous availability; no downtime
3. High velocity data (devices, sensors, etc.)
4. Data coming in from many locations
5. Structured data is available with some semi/unstructured data.
6. To maintain high data volumes; retain forever.

What Is NoSQL ?

- NoSQL systems are also called as "Not only SQL" which indicates that they may support query languages like SQL.
- NoSQL is not depending on column, rows or schema for structure, it is no-relational database management systems. The data models of NoSQL are more flexible.
- NoSQL provides scalability, availability and fault tolerance and emerges as a alternative for relational database.
- The speciality of NoSQL is that, it may not require fixed table schemas avoids join operations, and also scale horizontally.
- Developers are working with applications that create massive volumes of new, rapidly changing data types - structured, semi-structured, unstructured and polymorphic data. NoSQL is useful for data which is growing far more rapidly or unstructured data or data which does not store in the relational schemas of RDBMS. There are common types of unstructured data: user and session data; chat, messaging, and log data; time series data such as IoT and device data; and large objects such as video and images.

Features of NoSQL

- Design simplicity.
- Simpler "horizontal" scaling to clusters of machines. This was a problem in relational databases.
- More control over data availability.

Observations regarding NoSQL

- It does not use the relational model.
- It runs well on clusters.
- NoSQL is Mostly open-source.
- It is Schema-less.

Why NoSQL ?

- All IT professionals and industry database experts come to know that NoSQL is here to stay.
- A recent study performed on NoSQL market growth forecasts a very strong compound annual growth rate of 21 percent for NoSQL technology from 2013-2018. It shows bright future for NoSQL.

- NoSQL databases has been much more clearly articulated today. NoSQL database refers to groups of databases that are not based on relational database model.
- The data storage model used by NoSQL database is not some fixed data model, but the common features among the NoSQL database is that the relational and tabular database model of SQL based database is not used.

Advantages of NoSQL

We cannot consider that NoSQL databases are straight substitution for relational database management system (RDBMS). But for many issues regarding data, NoSQL seem to be better.

1. **Data storage :** NoSQL databases supports storing data "in the form of Key value pair which give ability to store simple data structures. The document NoSQL database provides the ability to handle a range of flat or nested structures.
2. **Support for unstructured text :** NoSQL databases can handle unstructured text easily. This ability increases information effectively and can help organizations make better decisions.
3. **Ability to handle change over time :** NoSQL databases are capable of managing changes because of the systematic storage system.
4. **No reliance on SQL magic :** SQL(Structured Query Language) is the predominant language which is used to write queries in relational database management systems. Even if several NoSQL databases provide support for SQL access, they do so for compatibility with existing applications like business intelligence (BI) tools. NoSQL is not dependent on SQL for processing. NoSQL databases support their own query languages that can support data processing.
5. **Ability to scale horizontally on commodity hardware :** NoSQL databases supports distribution of a database across several servers. Hence if there is requirement of more data storage, then number of servers can be increased and connect them to database cluster (horizontal scaling) making them work as a single data service.
6. **Breadth of functionality :** Near about all the relational databases support the same characteristics but in a slightly different way, so they are all similar. In contrast, the NoSQL databases come in different core types: key-value, document store and graph. Out of these types, the one select to suit our requirements is not hard.
7. **Support for multiple data structures :** There is requirement of simple as well as complex data structures. NoSQL databases provide support for a range of data structures. Key-value stores can handle Simple binary values, lists, maps, and strings. Document databases can manage highly complex parent-child hierarchal structures Graph stores can describe the web of interrelated information.
8. **Big data applications :** In some systems the data grows rapidly. Such big volume data can be easily handled by NoSQL databases.
9. **Database administration :** The NoSQL has data distribution and auto repair capabilities, simplified data models and fewer tuning and administration requirements. This leads to less requirement of hands-on management.
10. **Economy :** These databases are designed to be used with low-cost commodity hardware.

It is difficult for application developer to find match between the relational data structures and the in-memory data structures. Using NoSQL databases they can develop the system without having to convert in-memory structures to relational structures.

Disadvantages of NoSQL

1. No standard schema.
2. Less use of SQL.

Companies using NoSQL

Now a day's many companies using NoSQL. Some of them are :

- Google
- Facebook
- Adobe
- Foursquare
- Digg
- Vermont public radio
- LinkedIn
- Mozilla

Syllabus Topic : Types and Examples of NoSQL Database

6.2 Types and Examples of NoSQL Database

There have been various approaches to classify NoSQL databases, each with different categories and subcategories.

Following are some types of NoSQL :

1. Key Value Store
2. Document Store
3. Column Store
4. Graph

Syllabus Topic : Key Value Store

6.2.1 Key Value Store

SPPU - Oct. 16

University Question

Q. Explain key value store NOSQL data model. **(Oct. 2016(In sem), 5 Marks)**

- The Key-value (KV) store system uses the concept of associative array, as their fundamental data model. In this model, data is represented as a collection of key-value pairs. Every single item in the database is stored as an attribute name (or 'key'), together with its value.
- In NoSQL database, a table exists with two columns : one is the Key and the other is Value.
- The key in the key-value pair should not be repeated because it the unique identifier that helps to access the value associated with that key uniquely.
- The Key value stores allow the developer of application to store schema-less data.
- Key-value databases are the simplest form of the NoSQL databases. Other advanced models are mostly extensions to this key-value model.
- **Examples** include Memcached, Riak, ArangoDB, InfinityDB, Oracle NoSQL Database, Redis and dbm.
- All key-value databases are not similar; there are major differences between these databases. For example: Data in **MemcacheDB** is not persistent while in **Riak** it is persistant. Such features are useful when implementing some solutions. It is important to not only choose a key-value database based on your requirements, it is also important to choose which key-value database.
- Examples of key-value NoSQL database applications :
  - Dynamo
  - Redis
  - FairCom
  - MemcacheDB
  - Riak
- Four main core operations perform on key-value store :
  1. **Get(key) :** It return the single value of given key.
  2. **Put(key,value) :** It assigns value to key.
  3. **Multi-get(key1,key2,key3,..,keyn):** It returns multiple values of given multiple keys.
  4. **Delete(key) :** It deletes both key and value present in it.

Example

This is a simple example for key-value store. Here keys are the names of employees and values are their contact numbers.

Key	Value
Sam	(234) 567-8901
jack	(134)526-6845
Ron	(245)452-4584
kenny	(356)584-1458

Where key value store is used ?

Key-value databases can be used in many scenarios. Such as,

- **General Web / Computers**
  - User profiles
  - Article/blog comments
  - Emails
- **E-commerce**
  - Shopping cart contents
  - Product categories
  - Product details

Advantages of Key Value Store

- Key value store is the simplest type of NoSQL.
- Supports simple queries very efficiently.

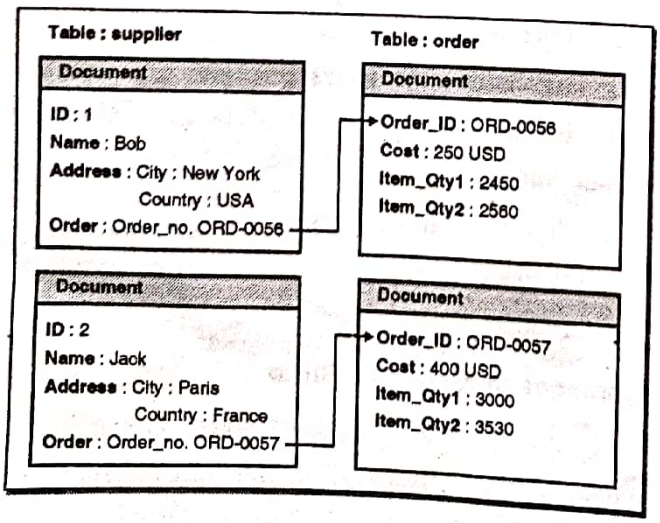
- Extended form of key-value stores is able to sort the keys.
- It is specially designed for storing data as a schema free data.
- Very simple data-modeling pattern should be understandable by anyone.
- With little or no maintained indexes, the key-value stores are designed to be more scalable and extremely fast.
- Suitable for system where data is not highly related.

Disadvantages of Key Value Store

- The indexing and scanning capabilities are absent. It does not help if we want to perform more operation as per user requirement than the basic CRUD (Create, Read, Update, Delete) operations.
- Only one row simple queries can be executed efficiently.
- Difficult to perform SQL operations like JOINS, GROUP BY etc.
- Selecting appropriate data type for value is difficult.
- Difficult to use constraints like FOREIGN KEY or NOT NULL.
- More application code is required to reassemble collections of key-value pairs into objects.

Syllabus Topic : Document Store

6.2.2 Document Store



- These databases store records as "documents" where a document can generally be thought of as a grouping of key-value pairs.
- The documents are identified by the unique keys which represents them. One defining characteristics of a document-oriented database is that in addition to the key lookup performed by a key-value store, the database offers an API or query language that retrieves documents based on their contents.
- Document databases are extension to key-value store. Addition to query capabilities of key-value databases, they provide indexing and the ability to filter documents based on attributes in the document.
- Examples of Document Store NoSQL database applications :
  - MongoDB
  - Couchbase

Advantages of Document Store

- The performance is good and the distribution across various servers becomes a lot easier.
- There is no need of translation between object in SQL and application. The object can directly be converted into document.
- They have strong indexing features and can rapidly execute different queries.

6.2.3 Column Store

- Column store is column oriented NoSQL database. Data is stored in cells grouped in columns of data rather than as rows of data. The logical grouping of columns is created in column families. There is no limitation on number of columns in a column family.
- These columns can be created at runtime. The operations like read write are performed using columns rather than rows.
- The column store structure gives the advantage of fast search / access and data aggregation over the row format data storage of relational databases.
- Relational databases store a single row as a continuous disk entry. Different rows are stored in different places on disk while column store database store all the cells related to a column as a

continuous disk entry which makes the search/access faster.

- **For example :** Displaying titles from a bunch of a million articles will be a tedious and time wasting task while using relational databases as it will go over each location to get item titles. While in column store, title of all the items can be obtained with just one disk access.
- Examples of column store NoSQL database applications :
  - HBase
  - BgTable
  - HyperTable

Advantages of Column Store

- Efficient storage and data compression.
- Fast data loads.
- Simple configuration.

Disadvantages of Column Store

- Queries with table joins can reduce high performance.
- Transactions are to be avoided or just not supported.

Syllabus Topic : Graph

6.2.4 Graph Store

- The Graph Store NoSQL database technology is designed to handle very large sets of data which may be structured, semi-structured or unstructured.
- In a Graph Base NoSQL Database, rigid format of SQL or the tables and columns representation does not exists. Rather a flexible graphical representation is used which is perfect to address scalability concerns. The graph structure contains edges, nodes and properties.
- The graph store is helpful in transferring the data from one model to other very easily.
- Graph store stores the entities and relationships between these entities. Entities are also known as nodes, which have properties. Nodes represent entities such as people, businesses, accounts, or any other item to be tracked.
- They are roughly the equivalent of the record, relation, or row in a relational database, or the document in a document database.
- Relations are known as edges that can have properties.

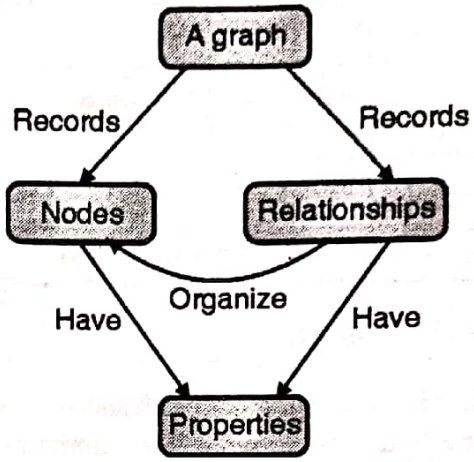


Fig. 6.2.1

- Edges are denoting directions. Nodes are generally organized with the help of relationships. The data is stored once by the organization of graph. This data can be interpreted in different ways depending upon the relationships between nodes.
- Key points related to graph store.
- Edges and nodes are used by these databases to represent and store data.
- The nodes are organised with the help of some relationships in between them, which is represented by edges between the nodes.
- Both the nodes and the relationships have some definite properties.
- Examples of Graph store NoSQL database applications :
  - Neo4j
  - Polyglot

Advantage of Graph Store

- Performance
- Flexibility
- Agility(ability to move quickly or easily)

Syllabus Topic : Distributed Database Model

6.5 Distributed Database Model

A database system can be viewed as consisting of three software modules: a transaction manager (TM), a data manager (DM), and a scheduler.

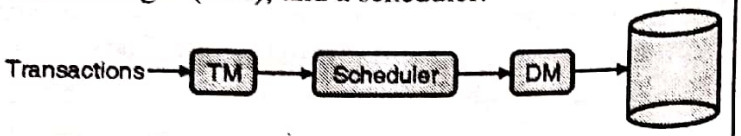


Fig. 6.5.1

- The transaction manager examines the execution of a transaction. It intercepts and executes all the transactions which have been submitted. Thus, the Transaction Manager is the mediator between users and the database system.
- Concurrency control is enforced by the scheduler. It grants or releases locks on data objects as per the requests of a transaction.
- The database is managed by the data manager. It performs the read-write requests issued by the transaction manager on behalf of a transaction by operating them on the database. The failure recovery is responsibility of data manager. Thus, the DM is the interface between the scheduler and the database.
- The concurrency control model of a distributed database system is shown in Fig. 6.5.2.

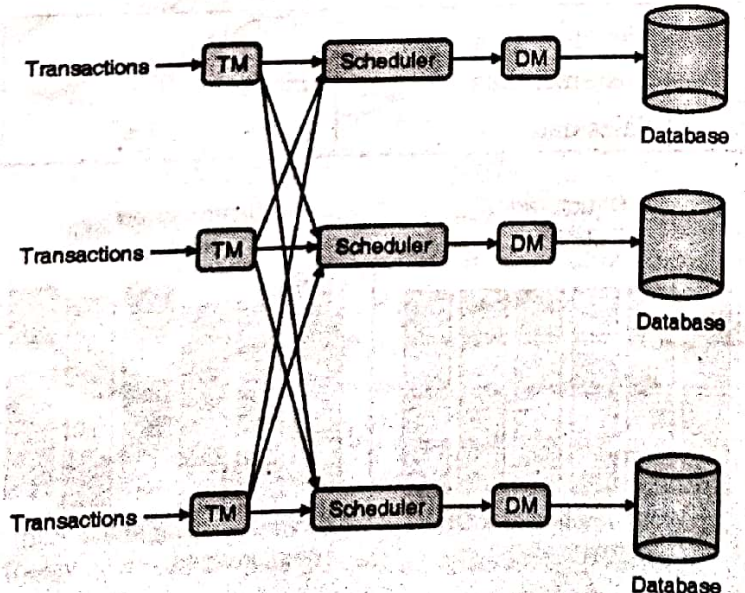


Fig. 6.5.2 : Distributed database system

Syllabus Topic : CAP Theorem and BASE Properties

6.6 CAP Theorem and BASE Properties

6.6.1 CAP Theorem

CAP stands for Consistency, Availability and Partitioning tolerance. In 2001 the CAP theorem was given by Eric Brewer, a professor at the University of California, Berkeley and one of the founders of Google, in the keynote of Principles of Distributed Computing.

Statement

- In general it is expected that every system should have Consistency, High-Availability and Partition-tolerance. But it is really hard for any system to achieve all these three properties at the same time.

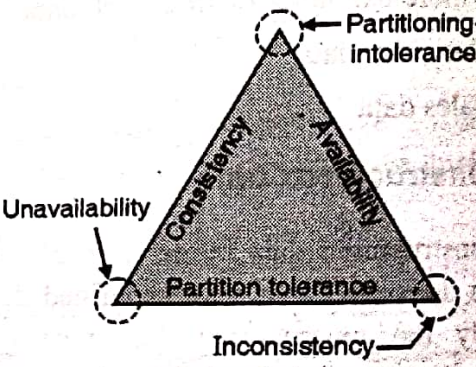


Fig. 6.6.1

- Maximum of two properties out of three can be achieved by the system. First we will see these three properties.

Consistency

- The system should follow the rule of sequence of updates which is present across all replicas in a cluster.
- Regardless of the location the data should be consistent. For example if client1 writes data in sequence A, B then another clients should be able to read the data in same order written by client1. It is also known as strong consistency.

Availability

- A service should be available to operate fully. It should be guaranteed that every request receives a

response about the status (successful or failed). It is also possible that a system which is not available may be consistent. It is hard to achieve *consistency* and *availability* at the same time.

- Both are contradictory to each other, i.e. if consistency is relaxed then system is highly available under the partitioning conditions (see next definition) and *strong consistency* means that in some specific conditions the system will not be available.

Partition Tolerance

- If failure may occur in any part of the system, the entire system does not get shut down.
- For example, consider cluster of n replicated nodes and network is unavailable among some number of nodes because of some reasons like network cable got chopped. Because of this synchronization of data is not possible.
- Here some part of system is in working condition while the other is not. If there is partition in the network, then there is possibility to lose consistency as we allow updates to both sides of the partition. Or we lose availability as we may shut down the system because of error until condition is resolved.
- A simple meaning of this theorem is "It is impossible for a protocol to guarantee both consistency and availability in a partition prone distributed system". This was mentioned above in example.

Adjustment between Requirements

1. **Available and Partition-Tolerant :** You have two nodes and the link between the two is severed. Since both nodes are up, the system can be designed to accept requests on each of the nodes. This insures the availability of system even if the network is partitioned. However independent results are issued by each node. Here high availability and partition tolerance is provided by compromising consistency.
2. **Consistent and Partition-Tolerant :** You have three nodes and one node loses its link with the other two. A rule cab made that a result will be returned only when a majority of nodes agree. So, even if the partition is there, the system will return a consistent result. Although the separated node is up, it won't be available since it is not able to reach consensus.
3. Finally, a system can be both *consistent and available*, but it possible that it may have to block on a partition. Most of the NoSQL database system architectures favour one factor over the other.

6.6.2 BASE Properties

SPPU - May 16

University Question

Q. Explain BASE properties of NOSQL database with suitable example. **(May 2016, 5 Marks)**

- In the Chapter 4, we have seen the ACID properties of DBMS. ACID properties are an important concept for databases.
- Let's recall in brief what ACID means in traditional RDBMS community before moving to the BASE Properties.

The four properties are as follows.

- **Atomicity :** This property states that, either all operations contained by a transaction are done successfully or none of them complete at all.
- **Consistency :** The consistency property ensures that the transaction executed on the database system will bring the database from one valid state to another.
- **Isolation :** In case of concurrent transactions, the isolation property ensures that the system state should be same that would be obtained if transactions were executed sequentially.
- **Durability :** The durability property assures that after transaction committed successfully the updates made should remain permanent in the database even in the event of power loss, crashes, or errors.

ACID provides *strong consistency* (synchronous transactions) for partitioned databases and thus provides less availability.

Consistency is always preferable, but it is not always available.

- **Base** Property means **Basically Available, Soft state, Eventual consistency.**
- Consistency is always preferable, but it is not always available. In the NoSQL world, ACID transactions are less suitable as some DBMS do not have requirements for strict consistency, data freshness and accuracy in order to gain other benefits, like scale and resilience. By considering this, the BASE Consistency model is developed.

- The BASE properties are as follows :
  1. **Basic Availability :** Most of the time the database appears to work.
  2. **Soft-state :** It is not necessary that the stores should be write-consistent. Also the different replicas have to be mutually consistent all the time.
  3. **Eventual consistency :** Stores may show the consistency at some later point. Eventual consistency which is normally asynchronous in nature is a form of a weaker consistency which improves speed and availability.
- The BASE (Basically Available, Soft state, Eventual consistency) is the opposite of ACID.
- ACID is pessimistic i.e. consistency is required at the end of every operation. BASE is optimistic, i.e. it accepts that there is uncertainty in consistency.
- A BASE Model focus on availability as it is important for scale, but it doesn't offer guaranteed consistency of replicated data at write time.
- At the end we can say the BASE model of consistency provides a less strict assurance than **ACID :** Data will be consistent in the future, either at read time or may be always consistent for some points.

Syllabus Topic : Comparative Study of SQL and NoSQL

6.7 Comparative Study of SQL and NoSQL

For managing the database SQL has been most widely used programming language over the last few decades. It is a Relational Database Management System. But in today's era NoSQL has arises for an option to the SQL. There is very high difference between SQL and NoSQL. In this topic we will see the comparative study of SQL and NoSQL.

The conceptual difference is :

A framework of a relational database which is setup with the defined categories used by SQL. The tables of SQL are idle for storing data which is structured. The structured data like name, address fits into the SQL format perfectly.

But when data is unstructured it needed another format which is not dependant on the relationships of the data. When this situation occurs that time we can use NoSQL. NoSQL allows for the storage of an unstructured data without categorizing the data into fixed tables. NoSQL database scales horizontally. NoSQL is also known as Non-relational database or distributed database.

Factual difference is

- In SQL databases are structured in the form of tables, but in NoSQL databases are structured in the form of documents, graphs, or key-value pairs.
- In SQL Database there is a standard definition of schema which must be worked with the structured data. While In NoSQL there is no standard definition for schema which must be worked with the structured data.
- SQL database have predefined schema for structured data while NoSQL database have dynamic schema for unstructured data.
- SQL databases has feature of vertical scaling while NoSQL databases has feature of horizontal scaling.
- SQL database are designed and managed with SQL (Structured Query language) while NoSQL database are designed and managed with the UnQL (Unstructured Query Language).
- The syntax of SQL does not vary with database, while the syntax of UnQL varies with database.
- SQL is preferable for handling complex query while NoSQL cannot handle complex query. SQL queries are more powerful than NoSQL queries.
- Examples of SQL databases are : MySQL, Oracle, MS-SQL while examples of NoSQL are : MongoDB, BigTable, Redis, Hbase, Neo4i and CouchDb.
- SQL cannot manage big data which is stored in hierarchical manner while NoSQL handles hierarchical data better than SQL. Hence, NoSQL is preferable to SQL when managing big data.

Comparision between SQL and NoSQL

Sr. No.	SQL	NoSQL
1.	SQL means structured query language.	NoSQL means NOT only SQL language.
2.	Data is in structured and organized form.	Data is in unstructured form.
3.	It is used for small to medium scale data set effectively.	It is used for large set of data.
4.	SQL is schema based.	NoSQL is schema less.
5.	Data is stored as row and column in table where each column is of specific type.	The data model is depending upon the database type.
6.	Relational database is table based.	Key value pair, storage, column, document store, graph database.
7.	Example : SQL server Oracle	Example : MongoDB HBase

Syllabus Topic : NoSQL Data Models

6.8 NoSQL Data Models

Data Model

Data model is the mode which organizes the data. It is a representation which is used to understand and manipulate the data. The data model helps to :

- Represent the data elements in analytical view.
- Maintain relationship of these elements.
- Describe the way by which we interact with the database.

In RDBMS the relational model was used to represent the data. The data is represented in the form of tables (combination of rows and columns). Each row represents some entity while columns represent relationships in the same table or with another table. The relational data modeling has been a well-defined discipline for many years. However now a days with the emergence of NoSQL databases, need of a new data model arises.

NoSQL Data Model

The NoSQL data model is different from traditional relational data model. There are different data models :

- Key-value
- Document
- Column - family
- Graph

The three model Key-value, document and column- family share common features of Aggregate Orientation.

NoSQL Agreegate Model

- In this model it considered that we have to manage more complex data compared to relational model. The complex data structure are in the form of map, list etc. The aggregate model uses this complex structure. Aggregate is a collection of data objects which are treated as a single unit to manage and manipulate. Atomic operations are expected to update aggregates. Using aggregate it is easy to work on cluster which is unit of machines. Aggregates helps application developer by solving the mismatch problem which usually occur in relational model.
- When the aggregate model runs on a cluster, it gives several advantages on computation power and data distribution. While gathering data, it requires minimizing the number of nodes. The aggregate gives an important view that which data should be stored together.
- The **Key-Value** and **Document** databases are strongly aggregate oriented. These databases contain number of aggregates with a key to get the data. In Key-Value we can store any type of object.
- The Column-Family has two level aggregate structure. The first key is the row identifier. The second level values are defined to as columns.

Important point related to NoSQL Data Aggregate Model

- All these models use aggregated index by a key.
- The key is used for searching the data.
- The Aggregate acts as a atomic unit for modification.

- In the document model, the document is treated as single unit of storage. This model makes document transparent for querying.
- In Column-Family model, the columns are divided into column families and treated as single units.
- All models improve the accessibility of data.

Syllabus Topic : Case Study - Unstructured Data from Social Media

Case study is in detail study of a phenomenon, like a person, group, or situation. The phenomenon is studied in detail, cases are analyzed and solutions or interpretations are presented.

6.9 Case Study - Unstructured Data from Social Media

In this case study we are going through following steps.

1. Background
2. Problem
3. Proposed Solution
4. Implementation
5. Results

1. Background

- The unstructured data is the form of information which does not have any predefined data model. This data is not suitable for traditional relational model. It is usually text-heavy, but also contains data like as dates, numbers etc.
- Such data is difficult to understand and manage using traditional programs because it generates irregularities and ambiguities.
- In social media maximum of data is in unstructured form. The most big data sources like WhatApp, Facebook, twitter have unstructured data. It is difficult to run any analytics on such data. To analyze such data we need to convert it into structured form.

2. Problem

For social media sites, it is really hard to organize such data. There are various problems to handle the data which lead to find out some solution. Social media contains different types of data some of which is important and some not.

- Chat History
- Posts of users
- Free advertisements posted by users
- Twits
- Comments

There is no any predefined or standard format of this data. Tweets, Facebook postings and other social comments have to be analyzed to determine the sentiment of the population. The business strategies are decided on this data.

3. Proposed Solution

To handle the unstructured data properly it should be converted into structured or semi structured format. But if volume of the data is very big then it is difficult to convert it. For such big data we can use the database framework like NoSQL and Software system like Hadoop.

There are two solutions for handling unstructured data :

A. Convert it into structured format.

B. Use database framework like NoSQL and Software system like Hadoop.

4. Implementation

A. Converting unstructured data into structured format

The unstructured data can be converted into semi-structured or structured data by converting the text into words and phrases which can fit into relational tables. Then it can be categorized. The analysis techniques can then be used to conclude and arrive at results. It is important that the type of data, the source and its general understanding has to be re-vamped. So does this process of converting unstructured to structured data may be manual or machine driven through algorithms. Algorithms reduce accuracy but increase scale.

B. Use database framework like NoSQL and Software system like Hadoop.

We have already seen the database framework NoSQL which can deal with such unstructured data. The database tools like Hadoop, MongoDB also are useful in dealing with unstructured data.

5. Results

Both above options are 100 percent effective in dealing with the vast social media unstructured data. Both options give a perfect solution to manage this data. Data management involves access and manipulation of data.

Syllabus Topic : Introduction to Big Data

6.10 Introduction to Big Data

Now a day the amount of data created by various advanced technologies like Social networking sites, E-commerce etc. is very large. It is really difficult to store such huge data by using the traditional data storage facilities.

Until 2003, the size of data produced was 5 billion gigabytes. If this data is stored in the form of disks it may fill an entire football field. In 2011, the same amount of data was created in every two days and in 2013 it was created in every ten minutes. This is really tremendous rate.

In this topic, we will discuss about big data on a fundamental level and define common concepts related to big data. We will also see in deep about some of the processes and technologies currently being used in this field.

Big Data

Big data means huge amount of data, it is a collection of large datasets that cannot be processed using traditional computing techniques. Big Data is complex and difficult to store, maintain or access in regular file system. Big Data becomes a complete subject, which involves different techniques, tools, and frameworks.

There are various sources of big data. Now a days in number of fields such huge data get created. Following are the some of fields.

- **Stock Exchange :** The data in the share market regarding information about prices and status details of shares of thousands of companies is very huge.
- **Social Media Data :** The data of social networking sites contains information about all the account holders, their posts, chat history, advertisements etc. On topmost sites like facebook and whatsapp, there are literally billions of users.
- **Video sharing portals :** Video sharing portals like youtube, Vimeo etc. contains millions of videos each of which require lot much of memory to store.
- **Search Engine Data :** The search engines like Google and Yahoo holds lot much of metadata regarding various sites.
- **Transport Data :** Transport data contains information about model, capacity, distance and availability of various vehicles.
- **Banking Data :** The big giants in banking domain like SBI or ICICI hold large amount of data regarding huge transactions of account holders.
- The data can be categorized in three types.
  1. **Structured Data :** This type of data is stored in relations(tables) in Relational Database Management System.
  2. **Semi-structured Data :** This type of data is neither raw data nor typed data in a conventional database system. A lot of data found on the web can be described as semi-structured data. This type of data do not have any standard formal model. This data is stored using various formats like XML and JSON.
  3. **Unstructured Data :** This is data do not have any pre-defined data model. The data of video, audio, Image, text, web logs, system logs etc. comes under this category.
- In general there are some important issues regarding data in traditional file storage system.
  1. **Volume :** Now a days the volume of data regarding different fields is high and potentially increasing day by day. Organizations collect data from a variety of sources, including business transactions, social media and information etc.
  2. **Velocity :** The configuration of system single processor, limited RAM and limited storage capacity system cannot store and manage high volume of data.
  3. **Variety :** The form of data from different sources is different.
  4. **Variability :** The flow of data coming from sources like social media is inconsistent

because of daily emerging new trends. It can show sudden increase in size of data which is difficult to manage.

5. **Complexity :** As the data is coming from various sources, it is difficult to link, match and transform such data across systems. It is necessary to connect and correlate relationships, hierarchies and multiple data linkages of the data.

- All these issues are solved by the new advanced **Big Data Technology.**

Big Data Technologies

- Big data technologies are based on the accuracy of analyzing the data. This leads to more operational efficiencies, reductions in cost, reduction in failure and data loss.
- To implement the Big Data Technology, there is requirement of strong infrastructure which can manage and process the big data. This data may be structured or unstructured. The infrastructure must be able to protect data privacy and security.

Characteristics of big data

- Big data stores and manages huge amount of data in time and cost effective manner.
- It can analyze data of any form like unstructured, structured or streaming.
- It can maintain multiple copies of the important data across clusters.
- The System Failure Mechanism of this technology is very strong which avoid the data loss.
- Data can be stored in blocks on different machines which can be merged any times as per requirement.
- It captures data from live events in real time.
- There are various technologies in the market from different vendors including Amazon, IBM, Microsoft, etc., to handle big data. There are two classes of Big Data technology which can handle big data.
  - Operational Big Data
  - Analytical Big Data

Operational Big Data

- It was developed to address the shortcoming of traditional database and it is faster and can deal with large quantity of data spread over multiple servers. We are also using cloud computing architectures to allow massive computation to run effectively as well as it is cost efficient. This has made Big Data workload easier to manage, faster to implement as well as cheaper. Here in addition to interaction with user it also provides artificial intelligence about the active data. For example in games the moves of user are studies and next course of actions are suggested.
- Systems like MongoDB and NoSQL comes under this category.
  - **MongoDB :** This system provides operational capabilities for interactive, real-time workloads where data is primarily captured and stored.
  - **NoSQL :** The new advanced cloud technology is used to implement the huge computations to be run efficiently and inexpensively. It helps to manage the data easily and fast. This Cloud Technology is used in NoSQL big data system.

Analytical Big Data

- These systems provides analytical capabilities for complex analysis which considers most or all of the data.
- For example Massively Parallel Processing (MPP) database systems and MapReduce.
- Analytical Big Data is addressed by MPP database systems and MapReduce. These technologies has evolved as a result of shortcoming in traditional database which deals with one servers only. On the other hand MapReduce provides new method of analyzing data which is beyond the scope of SQL.

Challenges regarding Big Data

There are various challenges relate to big data :

- Getting the source data
- Making correction in this data
- Storing the data
- Searching specific data depending upon some criteria
- Sharing data between machines and users
- Transferring data
- Analyzing the data

Syllabus Topic : HADOOP - HDFS, MapReduce

6.11 Hadoop

SPPU - Dec. 15

University Question

Q. What is Hadoop? **(Dec. 2015, 2 Marks)**

- Hadoop is an open source, Java-based programming framework which supports the processing and storage of extremely large sets of data in a distributed computing environment using simple programming models.
- Hadoop has very strong processing power and the ability to handle virtually unlimited parallel tasks.
- With the help of Hadoop, applications can be run on systems with thousands of commodity hardware nodes. It can handle thousands of terabytes of data. Hadoop has distributed file system which facilitates rapid data transfer rates among nodes. This allows the system to proceed even in case one or more nodes get failed. This approach avoids the unexpected data loss.
- Hadoop has quickly emerged as a foundation for big data processing tasks like scientific analytics of data, planning of business and sales, and processing enormous volumes of data including social media data.

History of Hadoop

- Computer scientists Doug Cutting and Mike Cafarella created Hadoop in 2006 to support distribution for the Nutch (search engine). The main aim is to increase the speed of search results by the distribution of data and implement calculations on different computers by multitasking.
- Later on Cutting joined Yahoo but him still works on the Nutch project with the ideas based on Google's early work with automating distributed data storage and processing.
- The Nutch Prject divided into two parts –
  - Nutch - Web crawler portion
  - Hadoop - Distributed computing and processing portion.
- In 2008, Yahoo released Hadoop as an open-source project. Now Apache Software Foundation (ASF) manages the Hadoop's framework and ecosystem of technologies.

6.11.1 Modules (Components) of Hadoop

SPPU - Dec. 14, Dec. 15, May 16, Dec. 16

University Questions

Q. Explain different components of HADOOP. **(Dec. 2014, Dec. 2015 7 Marks)**

Q. Explain in brief different building blocks of HADOOP. **(May 2016, Dec. 2016, 7 Marks)**

- **HDFS : HDFS** stands for Hadoop Distributed File System. It states that the files will be broken into blocks and stored in nodes over the distributed architecture. It provides high-throughput access to application data.

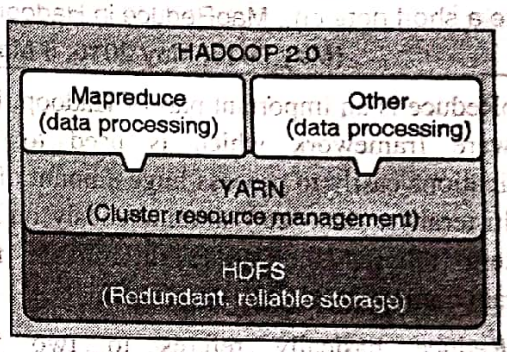


Fig. 6.11.1

- **Yarn :** Yarn stands for "Yet another Resource Negotiator". It is used for job scheduling and managing the cluster (multiple nodes).
- **Map Reduce :** This is YARN-based system for parallel processing of large data set using key value pair. The Map task takes input data and converts it into a data set which can be computed in Key value pair.
- **Hadoop Common :** These Java libraries and utilities are used to start Hadoop. These are used by other Hadoop modules. These libraries provide file system and OS level abstractions.

Advantages of Hadoop

1. Huge amounts of any kind of data can be stored and processed quickly.
2. **Computing power :** Hadoop's distributed computing model processes big data fast.
3. **Fault Tolerance :** In case of failure of any node the tasks are automatically redirected to other nodes.

4. **Flexibility :** Any kind of unstructured data like text, images and videos can be stored.
5. **Low cost :** This open-source framework is free.
6. **Scalability :** New nodes can be easily added to handle big tasks.

Disadvantage of Hadoop

1. Hadoop is rough in manner because the software is under active development.
2. Programming model is very restrictive.
3. Joins of multiple datasets are tricky and slow.
4. Cluster management is hard : In the cluster, operations like debugging, distributing. software, collection logs etc are too hard.
5. Requires care and may limit scaling.

6.11.1.1 MapReduce SPPU - May 15, May 16

University Question

Q. Write a short note on : MapReduce in Hadoop. (May 2015, May 2016, 5 Marks)

- MapReduce is an important part of Hadoop. It is a software framework which is used to write applications easily to process huge amount of data (multi-terabyte data-sets) simultaneously on large clusters (thousands of nodes) in reliable, fault-tolerant manner.
- MapReduce basically referes to two tasks performed by the Hadoop programs. One is map and another is reduce.

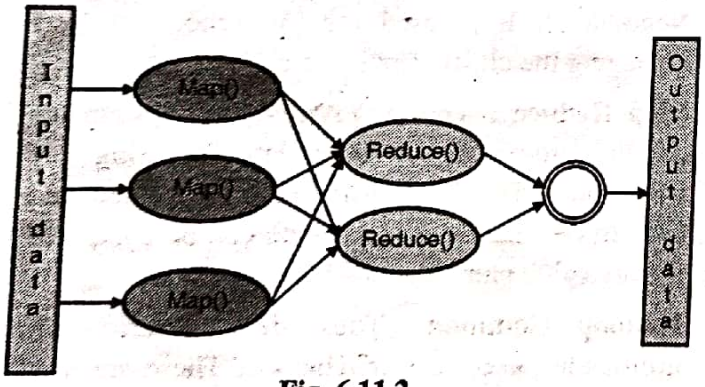


Fig. 6.11.2

- Hadoop programs perform following two tasks on **MapReduce :**
  - **The Map Task : This is the first task, which takes a set of data and converts it into another set of data in which individual elements are broken down into tuples (key/value pairs).**
  - **The reduce :** This job takes the output of previously executed map task as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.
- In MapReduce framework, the data of input and output is stored in a file system. The framework handles the scheduling of all the tasks, monitoring these tasks and if fails, re-executes them.
- The main advantage of MapReduce is that it is simple to scale data processing over multiple computing nodes. The data processing primitives are known as mappers and reducers in the MapReduce model.
- The MapReduce framework consists of single master JobTracker and one slave TaskTracker per cluster-node.
- **Master JobTracker :** The tasks under master are -
  - Managing the resources.
  - Tracking consumption and availability of resources.
  - Scheduling the jobs component tasks on the slaves.
  - Monitoring the tasks and re-executing the failed tasks.
- **The slaves TaskTracker :** It execute the tasks as per the directions of the master and provide task-status information to the master periodically.
- The JobTracker is very important in Hadoop MapReduce service. If JobTracker goes down, all running tasks get halted.

Advantages of MapReduce

1. Scalable.
2. Fault tolerant.
3. Simple coding model.
4. Supports unstructured data.

6.11.1.2 Hadoop Distributed File System (HDFS)

- The HDFS is the primary storage system used by Hadoop applications. HDFS is a distributed file system and a framework provided by Hadoop for the analysis and transformation of huge data sets which uses the MapReduce paradigm. The HDFS is based on Google File System (GFS). It provides

high-performance access to data across Hadoop clusters (thousands of computers), HDFS has become a key tool for managing pools of big data and supporting big data analytics applications.

- HDFS is usually deployed on commodity hardware of low-cost where the possibility of server failures is common. The file system is designed to be highly fault-tolerant. The HDFS facilitates the rapid transfer of data between different computer nodes and enables Hadoop systems to proceed its execution even if one or more nodes get failed. That decreases the risk of catastrophic failure, even in the event that numerous nodes fail.

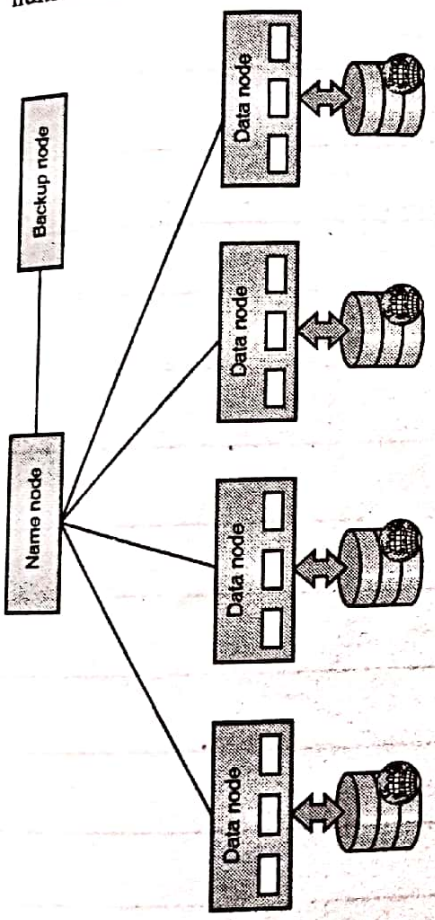


Fig. 6.11.3

- The architecture used by HDFS is known as master/slave architecture.
- NameNode which manages the metadata of file system and DataNode which stores the actual data.
- The HDFS namespace is a hierarchy of files and directories. Inodes are used to represent these file and directories. Inodes are used to record attributes such as permissions, modification and access times etc. The file content is split into large blocks and each block of the file is independently replicated at multiple DataNodes.
- The tree structure of namespace is maintained by the NameNode. It maps the blocks to DataNodes. In a cluster there may be hundreds of DataNodes and thousands of HDFS clients per cluster, as number of application tasks can be executed by each DataNode simultaneously.

Advantages of HDFS

1. High scalability.
2. Low limitation.
3. Open source.
4. Low cost.

Disadvantages of HDFS

1. Still rough - means software under active development.
2. Programming model is very restrictive.
3. Cluster management is high.