Unit – 1

Query Processing

- A Query processing in a distributed database management system requires the transmission of data between the computers in a network.
- A distribution strategy for a query is the ordering of data transmissions and local data processing in a database system.
- Generally, a query in Distributed DBMS requires data from multiple sites, and this need for data from different sites is called the transmission of data that causes communication costs.
- Query processing in DBMS is different from query processing in centralized DBMS due to this communication cost of data transfer over the network.
- The transmission cost is low when sites are connected through high-speed Networks and is quite significant in other networks.

1. Costs (Transfer of data) of Distributed Query processing :

- In Distributed Query processing, the data transfer cost of distributed query processing means the cost of transferring intermediate files to other sites for processing and therefore the cost of transferring the ultimate result files to the location where that result's required.
- Let's say that a user sends a query to site S1, which requires data from its own and also from another site S2. Now, there are three strategies to process this query which are given below:
  - We can transfer the data from S2 to S1 and then process the query.
  - We can transfer the data from S1 to S2 and then process the query.
  - We can transfer the data from S1 and S2 to S3 and then process the query.
- So the choice depends on various factors like, the size of relations and the results, the communication cost between different sites, and at which the site result will be utilized.
- Commonly, the data transfer cost is calculated in terms of the size of the messages. By using the below formula, we can calculate the data transfer cost:
  - Data transfer cost = C * Size

Where C refers to the cost per byte of data transferring and Size is the no. of bytes transmitted.

2. Using Semi join in Distributed Query processing:

- The semi-join operation is used in distributed query processing to reduce the number of tuples in a table before transmitting it to another site.
- This reduction in the number of tuples reduces the number and the total size of the transmission that ultimately reducing the total cost of data transfer.
- Let's say that we have two tables R1, R2 on Site S1, and S2.
  - Now, we will forward the joining column of one table say R1 to the site where the other table say R2 is located. This column is joined with R2 at that site. The decision whether to reduce R1 or R2 can only be made after comparing the advantages of reducing R1 with that of reducing R2.

Thus, semi-join is a well-organized solution to reduce the transfer of data in distributed query processing.

Data Fragmentation

- Fragmentation is a process of disintegrating relations or tables into several partitions in multiple sites.
- It divides a database into various subtables and sub relations so that data can be distributed and stored efficiently.
- Database Fragmentation can be of two types: horizontal or vertical.
- In a horizontal fragmentation, each tuple of a relation r is assigned to one or more fragments.
- In vertical fragmentation, the schema for a relation r is split into numerous smaller schemas with a common candidate key and a special attribute.

Methods of Data Fragmentation of a Table

Fragmentation is the task of dividing a table into a set of smaller tables. The subsets of the table are called fragments. Fragmentation can be of three types:

- **Horizontal Fragmentation**
  - In vertical fragmentation, the fields or columns of a table are grouped into fragments. In order to maintain reconstructiveness, each fragment should contain the primary key field(s) of the table. Vertical fragmentation can be used to enforce privacy of data.
- **Vertical Fragmentation**
  - Horizontal fragmentation groups the tuples of a table in accordance to values of one or more fields. Horizontal fragmentation should also confirm to the rule of reconstructiveness. Each horizontal fragment must have all columns of the original base table.
- **Hybrid Fragmentation** (combination of horizontal and vertical).
  - In hybrid fragmentation, a combination of horizontal and vertical fragmentation techniques are used. This is the most flexible fragmentation technique since it generates fragments with minimal extraneous information. However, reconstruction of the original table is often and expensive task.
- Horizontal fragmentation can further be classified into two techniques: primary horizontal fragmentation and derived horizontal fragmentation.

Fragmentation should be done in a way so that the original table can be reconstructed from the fragments. This is needed so that the original table can be reconstructed from the fragments. whenever required. This requirement is called “reconstructiveness.”

Advantages of Fragmentation

- Since data is stored close to the site of usage, efficiency of the database system is increased.
- Local query optimization techniques are sufficient for most queries since data is locally available.
- Since irrelevant data is not available at the sites, security and privacy of the database system can be maintained.

Disadvantages of Fragmentation

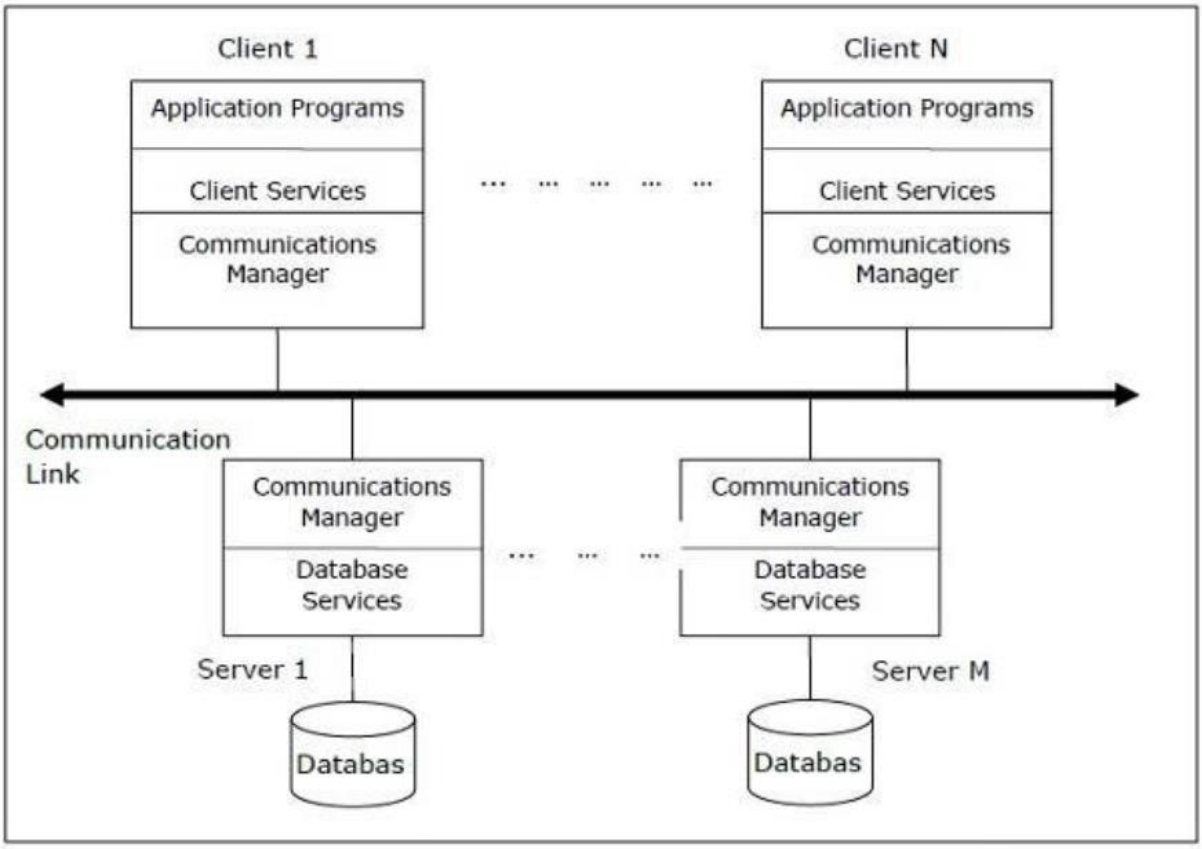
- When data from different fragments are required, the access speeds may be very low.
- In case of recursive fragmentations, the job of reconstruction will need expensive techniques.
- Lack of back-up copies of data in different sites may render the database ineffective in
- case of failure of a site.

Client Server Architecture for DDBMS

- This is a two level architecture where the functionality is divided into servers and clients
- The server functions primarily encompass data management, query processing, optimization and transaction management
- Client functions include mainly user interface
- However, they have some functions like consistency checking and transaction management

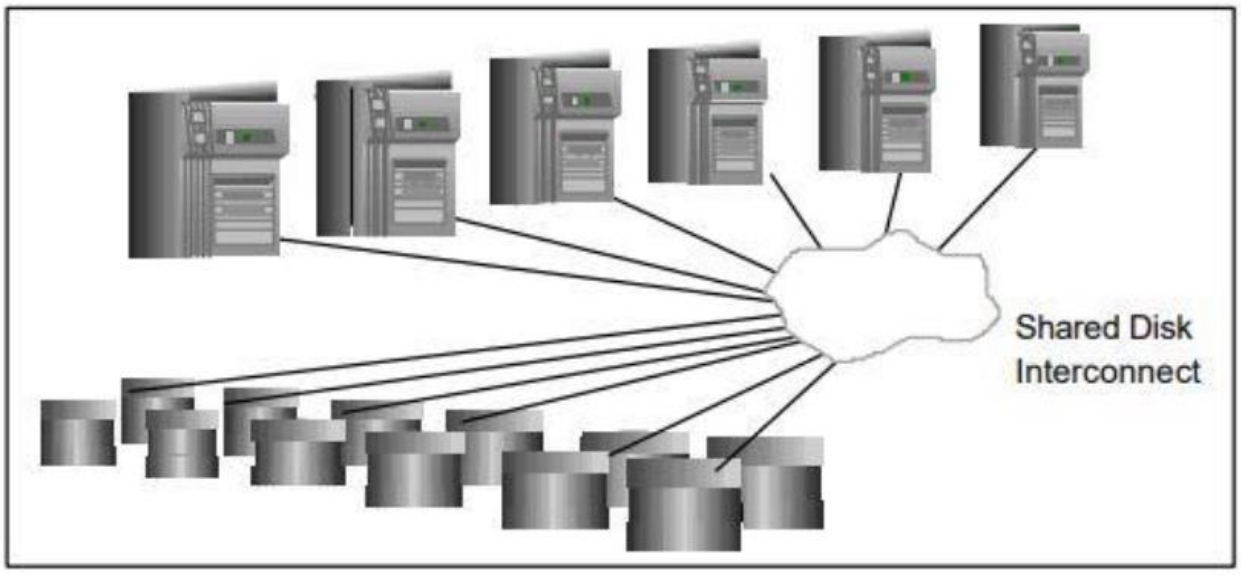
The two different client server architecture are

- Single Server Multiple Client
- Multiple Server Multiple Client



Cluster Database Architecture

A cluster consists of servers (usually SMPs), a cluster interconnect and a shared disk subsystem. Shared disk database architectures run on hardware clusters that give every participating server equal access to all disks – however, servers do not share memory. Most major hardware vendors provide shareddisk clusters today.



A database instance runs on every node of the cluster. Transactions running on any instance can read or update any part of the database there is no notion of data ownership by a node. System performance is based on the database effectively utilizing a fast interconnect, such as the Virtual Interface Architecture (VIA), between cluster nodes. Oracle9i Real Application Clusters (RAC) is the first successful shared-disk cluster architecture and utilizes sophisticated Cache Fusion ™ shared-cache algorithms to allow high performance and scalability without data or application partitioning.

Unit – 2

Distributed Concurrency Control

Concurrency controlling techniques ensure that multiple transactions are executed simultaneously while maintaining the ACID properties of the transactions and serializability in the schedules.

Concurrency control is provided in a database to:

(i) enforce isolation among transactions.
(ii) preserve database consistency through consistency preserving execution of transactions.
(iii) resolve read-write and write-read conflicts.

Locking Based Concurrency Control Protocols

Locking-based concurrency control protocols use the concept of locking data items. A **lock** is a variable associated with a data item that determines whether read/write operations can be performed on that data item.

One-phase Locking Protocol

In this method, each transaction locks an item before use and releases the lock as soon as it has finished using it. This locking method provides for maximum concurrency but does not always enforce serializability.

Two-phase Locking Protocol

- In this method, all locking operations precede the first lock-release or unlock operation. The transaction comprise of two phases.
- In the first phase, a transaction only acquires all the locks it needs and do not release any lock. This is called the expanding or the **growing phase**.
- In the second phase, the transaction releases the locks and cannot request any new locks. This is called the **shrinking phase**.
- The fundamental decision in distributed locking-based concurrency control algorithms is where and how the locks are maintained (usually called a lock table).

Centralized 2PL

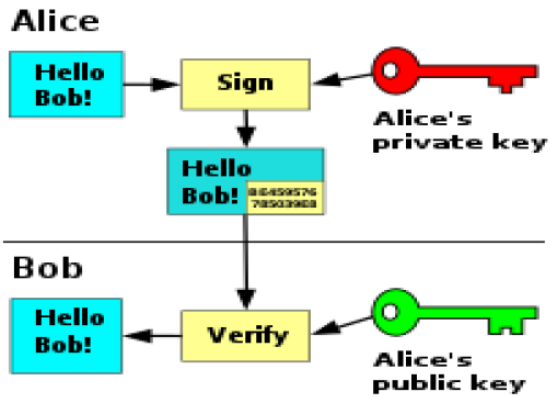
The 2PL algorithm can easily be extended to the distributed DBMS environment by delegating lock management responsibility to a single site. This means that only one of the sites has a lock manager; the transaction managers at the other sites communicate with it to obtain locks. This approach is also known as the **primary site 2PL algorithm**

Distributed 2PL

Distributed 2PL (D2PL) requires the availability of lock managers at each site.
The communication between cooperating sites that execute a transaction according tothe distributed 2PL protocol. The distributed 2PL transaction management algorithm is similar to the C2PLTM,with two major modifications.

Digital Signatures

- A **digital signature** is a mathematical scheme for verifying the authenticity of digital messages or documents.
- A valid digital signature, where the prerequisites are satisfied, gives a recipient very high confidence that the message was created by a known sender, and that the message was not altered in transit.
- Digital signatures are a standard element of most cryptographic protocol suites, and are commonly used for software distribution, financial transactions, contract management software, and in other cases where it is important to detect forgery or tampering.
- Digital signatures are often used to implement electronic signatures, which includes any electronic data that carries the intent of a signature, but not all electronic signatures use digital signatures.
- Digital signatures employ asymmetric cryptography.
- These provide a layer of validation and security to messages sent through a non-secure channel.
- Digital signatures are equivalent to traditional handwritten signatures in many respects, but properly implemented digital signatures are more difficult to forge than the handwritten type.
- Digital signature schemes, in the sense used here, are cryptographically based, and must be implemented properly to be effective. They can also provide non-repudiation, meaning that the signer cannot successfully claim they did not sign a message, while also claiming their private key remains secret.
- Further,some non-repudiation schemes offer a timestamp for the digital signature, so that even if the private key is exposed, the signature is valid.
- Digitally signed messages may be anything representable as a bitstring: examples include electronic mail, contracts, or a message sent via some other cryptographic protocol.



Alice signs a message—"Hello Bob!"—by appending a signature computed from the message and her private key. Bob receives both the message and signature. He uses Alice's public key to verify the authenticity of the signed message shown in Fig.

Unit – 3

Protocols Based upon Symmetric Cryptosystems

In a symmetric cryptosystem, knowing the shared key lets a principal encrypt and decrypt arbitrary messages. Without such knowledge, a principal cannot create the encrypted version of a message, or decrypt an encrypted message.

Basic Protocol
Using the above principle, we immediately obtain the basic protocol (shown in Algorithm) where principal P is authenticating itself to principal Q. "k" denotes a secret key that is shared between only P and Q.

Modified protocol with nonce
To prevent replay attacks, we modify the protocol by adding a challenge-and response step using a **nonce**. A nonce is a large random or pseudo-random number that is drawn from a large space so that it is difficult to guess by an intruder. This property of a nonce helps ensure that old communications cannot be reused in replay attacks.

Kerberos Authentication Service
Kerberos primarily addresses client–server authentication using a symmetric cryptosystem, together with trusted third-party authentication servers. (Project Athena)
The basic components include ***authentication servers* (*Kerberos servers*) and *ticket-granting servers* (TGSs).**

Protocols Based upon Asymmetric Cryptosystems

In an asymmetric cryptosystem, each principal P publishes its public key kp and keeps secret its private key kp−1. Thus only P can generate {m}kp−1 for any message m by signing it using kp−1 . The signed message {m}kp−1 can be verified by any principal with the knowledge of kp (assuming a commutative asymmetric cryptosystem).
Asymmetric authentication protocols can be constructed using a design principle called ASYM, which is as follows:
If a principal can correctly sign a message using the private key of the claimed identity, this act constitutes a sufficient proof of identity.

This ASYM principle follows the proof-by-knowledge principle for authentication, in that a principal's knowledge is indirectly demonstrated through its signing capability.

Basic Protocol
Q sends a random number n to P and challenges it to encrypt with its private key. P encrypts (P Q n) with its private key kp−1 and sends it to Q. Q verifies the received message by decrypting it with P's public key kp and checking with the identity of P, Q, and n.

A modified protocol with a Certification Authority
The basic protocol requires that Q has the knowledge of P's public key. A problem arises if Q does not know P's public key. This problem is alleviated by postulating a centralized certification authority (CA) that maintains a database of all published public keys. If a user A does not have the public key of another user B, A can request B's public key from the CA.

SSL Protocol

- The secure sockets layer (SSL) protocol was developed by Netscape and is the standard Internet protocol for secure communications.
- The secure hypertext transfer protocol (HTTPS) is a communications protocol designed to transfer encrypted information between computers over the World Wide Web.
- HTTPS is http using a secure socket layer (SSL). SSL resides between TCP/IP and upper-layer applications, requiring no changes to the application layer.
- SSL is used typically between server and client to secure the connection. One advantage of SSL is that it is application protocol independent.

The SSL protocol, in general, provides the following features:

- **End point authentication:** The server is the "real" party that a client wants to talk to, not someone faking the identity.
- **Message integrity:** If the data exchanged with the server has been modified along the way, it can be easily detected.
- **Confidentiality:** Data is encrypted. A hacker cannot read your information by simply looking at the packets on the network.