UNIT- V

1) What is Distributed Database?

A) A database that is distributed among a network of geographically separated locations. A distributed database is not entirely stored in one central location but is distributed among a network of locations that are geographically separated and connected by communication links. Each location has its own database and it also able to access data maintained at other locations.

The reasons for the development and use of distributed database systems are several and include the following:

Advantages: -

1. Often organizations have branches or divisions in different locations. For a given location, L, there may be a set of data that is used frequently perhaps exclusively, at L. In addition, L may sometimes need data that are used more frequently at another location, L.
2. Allowing each site to store and maintain its own database allows immediate and efficient access to data that are used most frequently. Such data may be used at others site as well, but usually with less frequency. Similarly, data stored at other locations can be accessed as required.
3. Distributed database can upgrade reliability. If one site's computer fails, or if a communication link goes down, the rest of the network can possibly continue functioning. Moreover when data are replicated at two or more sites, required data may still be available from a site, which is still operate.
4. Allowing local control over the data used most frequently at a site can improve user satisfaction with the database system. That is to say, local database can more nearly reflect an organization's administrative structure and thereby better service its manager's needs.

2) What is Distributed Database System Design?

A) The design of a distributed database system can be a complex task. Careful consideration must be given to the objectives and strategies to be served by the design and parallel decisions must be made as to how the data are to be distributed among the various network sites.

Strategies and Objectives: - Some of the strategies and objectives that are common to most implementation of distributed database systems are:

1) **Location Transparency: -** Location transparency enables a user to access data without knowing, or being concerned with, the site at which the data reside. The location of the data is hidden from the user.

2) **Replication Transparency: -** Replication transparency means that when more than one copy of the data exists, one copy must be chosen when retrieving data, and all copies must be updated when changes are made. Choosing one copy of the data for retrieval and always ensuring that all copies of the data are updated can be a burden on users. A DBMS should handle all such requirements, thereby freeing the user to concentrate on information needs.

3) **Configuration Independence: -** Configuration independence enables the organization to add or replace hardware without changing the existing software components of the DBMS.

Configuration independence results in a system that is expandable when its current hardware is saturated.

4) No homogeneous DBMSs: - It is sometimes desirable to integrate database maintained by different DBMS on different computers. Often the DBMS are supplied by different vendors and may support different data models. One approach to integrating this database is to provide a single user interface that can be used to access the data maintained by the non-homogenous DBMS. The different data models supported by the non-homogenous DDBMS are hidden from the user by this single, system wide interface.

5) Data Replication: - Data replication occurs if the system maintains several identical copies of a relation, R, with each copy being stored at a different site. Typically replication is introduced to increase the availability of the system: When a copy is unviable due to site failure(s), it should be possible to access another copy.

3) What is Distributed Query Processing?

A) Some database systems support relational databases whose parts are physically separated. Different relations might reside at different sites, multiple copies of a single relation can be distributed among several sites, or one relation might be partitioned into subrelations and these subrelations distributed. In order to evaluate a query posed at a given site, it may be necessary to transfer data between various sites. The key consideration here is that the time required to process such a query will largely be comprised of the time spent transmitting data between sites rather than the time spent on retrieval from secondary storage or computation.

Semijoin: - Suppose the relations R and S shown in Figure. Is stored at sites 1 and 2, respectively. If we wish to respond to a query at site 1 which requires the computation:

JOIN (R, S),

We could transmit all of S from site 2 to site 1 and compute the join at site 1. This would involve the transmission of all 24 values of S.

Site 1		Site 2		
R		S		
A1	A2	A2	A3	A4
1	3	3	13	16
1	4	3	14	16
1	6	7	13	17
2	3	10	14	16
2	6	10	15	17
3	7	11	15	16
3	8	11	15	16
3	9	12	15	16

Another way would be to compute

T = R [A2]

At site 1; then send T (6 values) to site 2, and compute

U = JOIN (T, S);

And finally send U (9 values) to site 1. We can then compute the desired

JOIN (R, S),

As
JOIN (R, U).

These steps and their results are shown in Figure 12.16. Note that with this approach we have only transmitted 15 values to complete the query.

This example provides a basis for defining a semijoin. The semijoin of R with S is

SEMIJOIN (R, S) = <a projection of those attributes of R that intersect those of S>,

Which is simply that portion of R that joins with S. Therefore,

JOIN (R, S) = JOIN (R, (SEMIJOIN (R, S), S)).

If R and S are at different sites, computing join (R, S) as shown previously saves transmitting data whenever R and S not join completely.

4) What is Distributed Two-Phase Locking?

A). Two-phase locking (2PL) synchronizes reads and writes by explicitly detecting and preventing conflicts between concurrent operations. Before reading date item x, a transaction must have a read lock on x. Before writing into x, it must have a write lock on x. The ownership of locks is generally governed by two rules.

1. Different transactions cannot simultaneously own conflicting locks,
2. Once a transaction surrenders ownership of a lock, it may never obtain additional locks.

5) What is a Cient/Sever system?

A). The term Client/Server (CS) involve multiple computers connected in a network is a concept of CS systems is that one or more of these computers may function as a provider of services to the remaining computers, which function as Clients that process applications. Client and Server have more formally been defined as follows:

Client: - A computer or workstation attached to a network that is used to access network resources.

Server: - A computer that furnishes clients with services such as database, connection to a network, or large disk drives. Server can be mainframes, minicomputers, large workstations or LAN devices. More than one server can be involved in providing services to clients.

6) What are the Functions and Capabilities of DBMS?

(Or)

What is DBMS Selection and Implementation?

A) The Data Dictionary/Directory: - An effective database system will allow growth and modification in the database without comprising the integrity of its data. The data dictionary/directory (DD/D) aids the accomplishment of this objective by allowing the definitions of data to be maintained separately from the data itself. This allows changes to be made to the data definitions with no effect on the stored data. For example, the subschema used by a particular program could be modified without in any way affecting the stored data. Other benefits provided by the DD/D include these:

- Physical storage structures can be changed without affecting the programs that use the data.
- Passwords and other security measures can be stored in the DD/D to facilitate control over data access.
- Centralized data definition enables easy reporting on the status of the database: Why is responsible for the various data items.

To yield these benefits, the DD/D usually includes the following features:

- A language for defining entries in the DD/D.
- A manipulation language for adding, deleting, and modifying entries in the DD/D
- Methods for validating entries in the DD/D
- Means for producing reports concerning the data contained in the DD/D.

Data Security and Integrity: -

1) Access Controls: - Access control is an important factor because they are a means of preventing unauthorized access to data. In the data-sharing database environment, good access controls are essential.

2) Concurrency controls: - Concurrency controls are a means of manipulating data integrity in the multi-user environment. Suppose user A and user B both access a given record at (essentially) the same time in order to process a transaction against the record. The DBMS must somehow limit access by one of the users until the others transaction has been completed. Without this type of facility, the accuracy and consistency of the database can rapidly erode.

3) **View Controls: -** It provides an automated means of limiting what a user is allowed to access from a given relation. This is a powerful feature that is commonly provided by relational DBMS. The ease of creating views and the capability of the view facility can be a useful distinguishing factor among DBMSs. The DBMS purchaser may also be interested in whether views can be updated and what limitations may apply.

4) **Encryption: -** It facilitates may be important to institutions whose databases contain very sensitive data. Encryption can also be important for the maintenance of a secure password directory.

5) **Backup and Recovery controls: -**Effective Backup and recovery controls are absolutely essential to efficient operation of the database system. The ease of use of backup and recovery controls, and their completeness, and their reliability should be major factors in the DBMS selection decision.

Query, Data Manipulation, and Reporting Capabilities: -

The DBMS's ability to support reporting requirements, along with users' query and data manipulation needs, is the cornerstone of today's management information systems. A sound

DBMS is going to provide the capability to generate structured reports in a variety of formats. In addition, the DBMS will provide a query language that is powerful, yet easy to learn and use. The language should be able to support both planned and unplanned query requirements with short response times.

Support of Specialized Programming Requirements: -

Developing specialized programs to interface with the DBMS requires facilities for supporting program development and program testing. A worthy DBMS will provide a host language for expressing standard procedural program structures or will provide an interface capability for quick prototyping of applications.

Physical Data Organization Options: -

The firm acquiring a DBMS may not wish to involve itself in the details of physical data organization. Instead, it may gauge the efficiency of a DBMS's physical organization by running sample applications.

For those who are interested, however, exploring the physical organization features may be of value. For example, it is known that the inverted list is most efficient in supporting multikey retrieval, whereas the chain list is superior for file updating since there is no need for updating a separate file. Information on other architectural features may be elicited in the process of considering the DBMS's capability to support the types of applications common to the firm.