I. Section-A: 5 X 4 =20 Marks

1. Evolution of Database

The sophistication of modern database technology is the result of a decades-long evolution in data processing and information management. Tugged on one side by the needs and demands of management and restrained on the other by the limits of technology, data access technology has developed from the primitive methods of fifties of the powerful, integrated systems of today.

Management's expectations have grown in parallel to the evolution of technology. The early data processing systems performed clericals tasks that reduced paper handling. More recent systems have expanded to production and management of information, which has come to be viewed as a vital company resource. Today the most important function of database systems is to provide the basis for corporate management information systems.

Implementation of technological change has been guided by genuine business needs. Management will only authorize a new computer system when it sees a clear benefit to offset the system's cost. And despite pitfalls and risks, benefits have been realized in many cases. Moreover, the end is not yet in sight and won't be for some time to come. Moreover, the end is not yet in sight and won't be for some time to come. New technology, such as object-oriented databases and client/server platforms, addresses new problems and will result in more powerful systems for the future.

The close relationship between database technology and business needs may be easier to understand if a closer look at the experience of International Product Distribution is taken into consideration.

DBMS were first introduced during the 1960s and have continued to evolve during subsequent decades.

1960s: Traditional file systems and first database management systems were introduced.

1970s: Hierarchical and Network data base models also known as first generation DBMS

1980s: Relational model also known as second generation DBMS. In Relational model, all data are represented in the form of tables. A relatively simple fourth generation language called SQL (for Structured Query Language) is used for data retrieval.

1990s:- Object-Oriented and Object-Relational data model

2. Primary Key

Primary Key: A primary key is the candidate key which is selected as the principal unique identifier. Example, empno in EMP relation, Studno in Student relation, Deptno in Dept table.

Primary Key is used to define with one column or set of columns that uniquely identify a Record and avoid duplicate Records of Table in the Relational Database. Primary Key must define with Unique and Not Null. Each Table can define with only one Primary Key.

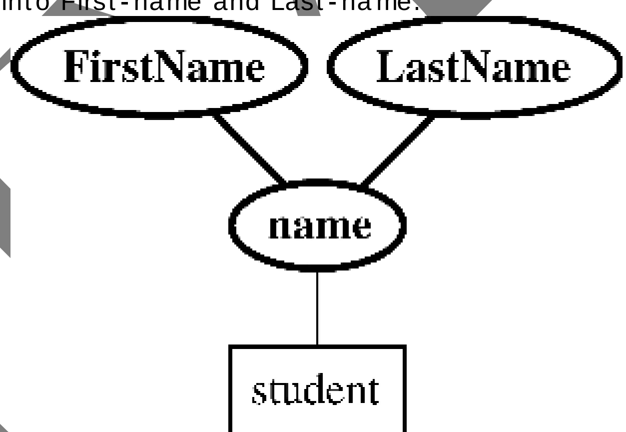
3. Arithmetic Operators in SQL.

SQL mathematical operations are performed using arithmetic operators (+, -, *, /, and %). We can use SQL like a calculator to get a feel for how these operators work. These are also combined with query for performing calculation for the selected data in the table. These operations are combined within update command in order perform calculation for the data.

Operator	Description	Example
+ (unary)	Makes operand positive	SELECT +3 FROM DUAL;
- (unary)	Negates operand	SELECT -4 FROM DUAL;
/	Division (numbers and dates)	SELECT SAL / 10 FROM EMP;
*	Multiplication	SELECT SAL * 5 FROM EMP;
+	Addition (numbers and dates)	SELECT SAL + 200 FROM EMP;
-	Subtraction (numbers and dates)	SELECT SAL - 100 FROM EMP;

4. Composite Attribute.

- Composite attribute: If an attribute can be split into components, it is called a composite attribute. Example for composite attribute: Name of the student can be splitted into First-name and Last-name.



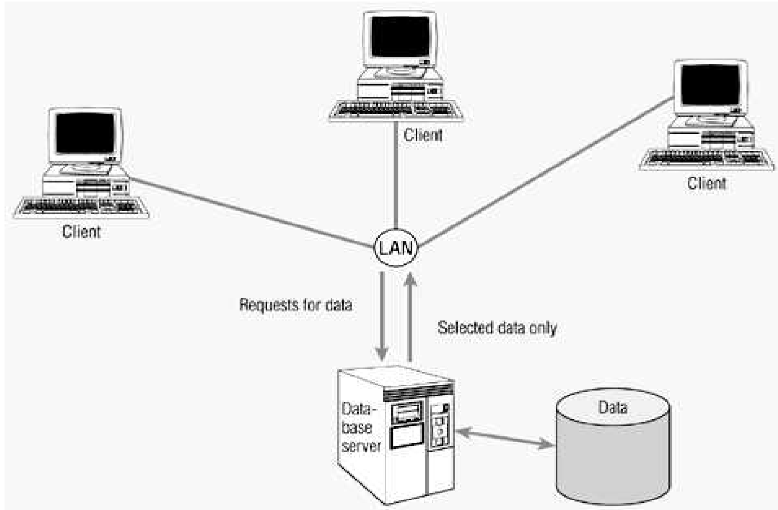
5. Two-Tier Architecture.

- Client is responsible for
  - I/O processing logic
  - Some business rules logic
- Server performs all data storage and access processing.
  - ➔ DBMS is only on server

Advantages of Two-Tier Approach:

- Clients do not have to be as powerful
- Greatly reduces data traffic on the network
- Improved data integrity since it is all processed centrally

Prasanth Kumar K
(Head-Dept of Computers, IIMC)



6. Data Independence.

Data Independence: The ability to modify a schema definition in one level without effecting a schema definition in the next level is called Data Independence.
There are two levels of data independence:
(i). Physical Data Independence: The ability to modify the physical schema without causing application programs to be rewritten.
(ii). Logical Data Independence: The ability to modify the conceptual schema without causing application programs to be rewritten.

Logical Data Independence is more difficult to achieve than physical data independence since application programs are heavily dependent on the logical structure of the data they access

7. Database Security.

database security methods focus on preventing unauthorized users from accessing the database. Because DBMS features that make the database easy to access and manipulate also open doors to intruders, most DBMS include security features that allow only authorized persons or processing that can be accompanied once access is made.

Authentication: Database access usually requires user authentication and authorization. For user authentication, the first level of security establishes that the person seeking system the user knows, such as log-on number and password, (2) something the user possesses, such as plastic ID card, or (3) a physical representation of the user, such as fingerprint or voiceprint.

Authorization and views : A view is a means of providing a user with a personalized model of the database. It is also a useful way of limiting a user's access to various positions of the database: Data a user does not need to be are simply hidden from view. This simplifies system usage while promoting security. Executing selects, projections, and joins on existing relations can represent views. The user might also be restricted from seeing any part of the existing relation or from executing joins on certain relations.

Types of Views: Different types of access authorization may be allowed for a particular view, such as the following:

- Read authorization: allows reading, but not modification of data.
- Insert authorization: allows insertion of new data, but no modification of existing data.
- Update authorization: allows modification of data, but not deletion.
- Delete authorization: allows deletion of data.

Views and security in SQL:
CREATE VIEW viewname (list of attributes desired, if different from base table)
As query.

Encryption: The various authentication and authorization measures that are standard for protection access to database may not be adequate for highly sensitive data. In such instances, it may be desirable to encrypt the data. Encrypted data cannot be read by an intruder unless that party knows the method of encryption. Considerable research has been devoted to developing encryption methods.

8. Data Integrity

A distributed database system differs from a centralized database system in that its database resides at a set of sites S. As might be expected, control data integrity becomes a harder problem in the network environment. Transactions are no longer linearly ordered sequences of actions on data.

The most common problem is when two (or more) transactions are executing at the same time, and both require access to the same data record in order to complete their processing. There may be multiple copies of the same record. All copies must have the same value at all times, or else transactions may operate on inaccurate data in DDB.
Most concurrency control algorithms for distributed database systems use some form of check to see that the result of a transaction is the same as if its actions were executed serially.
To implement concurrency control, the following must be known:

1. The type of scheduling algorithm used.
2. The location of the scheduler.
3. How replicated data are controlled.

When transactions are executed in parallel against the database, consideration of these issues by examining some of the principal methods of maintaining data integrity in a distributed database system by

1. Two phase commit protocol
2. Distributed locking
3. Time stamping

I. Section-B: 5 X 10 =50 Marks

9(a). Explain advantages of database approach.

The following are advantages of DBMS over File Oriented System:

(i). Data redundancy and Inconsistency: Since files and application programs are created by different programmers over a long period of time, the files are likely to be have different formats and the programs may be written in several programming languages. Moreover, the same piece of information may be duplicated in several places. This redundancy leads to higher storage and access cost. In addition, it may lead to data inconsistency, i.e. the various copies of same data may no longer agree.

(ii). Difficulty in accessing data: the conventional file processing environments do not allow needed data to be retrieved in a convenient and efficient manner. Better data retrieval system must be developed for general use.

(iii). Data isolation: Since data is scattered in various files, and files may be in different formats, it is difficult to write new application programs to retrieve the appropriate data.

(iv). Concurrent access anomalies: In order to improve the overall performance of the system and obtain a faster response time, many systems allow multiple users to update the data simultaneously. In such an environment, interaction of concurrent updates may result in inconsistent data.

(v). security problems: Not every user of the database system should be able to access all the data. For example, in banking system, payroll personnel need only that part of the database that has information about various bank employees. They do not need access to information about customer accounts. It is difficult to enforce such security constraints.

(vi). Integrity problems: The data values stored in the database must satisfy certain types of consistency constraints. For example, the balance of a bank account may never fall below a prescribed amount. These constraints are enforced in the system by adding appropriate code in the various application programs. When new constraints are added, it is difficult to change the programs to enforce them. The problem is compounded when constraints involve several data items for different files.

(vii). Atomicity problem: A computer system like any other mechanical or electrical device is subject to failure. In many applications, it is crucial to ensure that once a failure has occurred and has been detected, the data are restored to the consistent state existed prior to the failure.

9(b). Explain various phases in Database Development Life Cycle.

DDLC (Database Development Life Cycle):

It is a process for designing, implementing and maintaining a database system.

It consists of six stages:

1. Preliminary design
2. Feasibility design
3. Requirements definition
4. Conceptual design
5. Implementation
6. Database evaluation and maintenance.

Prasanth Kumar K
(Head-Dept of Computers, IIMC)

Preliminary Planning: It is a specific database system takes place during the strategic database planning project. After the database implementation project begins, the general information model produced during database planning is reviewed and enhanced if needed. During this process, the firm collects information to answer the following questions:

- How many application programs are in use, and what functions do they perform?
- What files are associated with each of these applications?
- What new applications and files are under development?

This information can be used to establish relationships between current applications and to identify uses of application information. It also helps to identify future system requirements and to assess the economic benefits of a database system.

Feasibility Study: A feasibility study involves preparing report on the following issues:

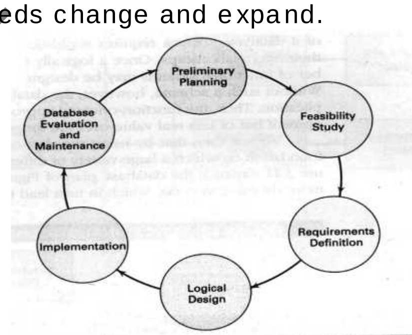
1. Technological feasibility: Is suitable technology available to support database development?
2. Operational feasibility: Does the company have personnel, budget and internal expertise to make a database system successful?
3. Economic feasibility: Can benefits be identified? Will the desired system be cost-beneficial? Can costs and benefits be measured?

Requirements Definition: It involves defining the scope of the database identifying management and functional area information requirements and establishing hardware/software requirements. Information requirements are determined from questionnaire responses, interviews with managers and clerical users and reports and forms currently being used.

Conceptual Design: The conceptual design stage creates the conceptual schema for the database. Specifications are developed to the point where implementation can begin. During this stage, detailed models of user view are created and integrated into a conceptual data model recording all corporate data elements to be maintained in the database.

Implementation: During database implementation, a DBMS is selected and acquired. Then the detailed conceptual model is converted to the implementation model of the DBMS, the data dictionary built, the database populate, application programs developed and users trained.

Database Evaluation & Maintenance: Evaluation involves interviewing users to determine if any data needs are unmet. Changes are made as needed. Over time the system is maintained via the introduction of enhancements and addition of new programs and data elements as business needs change and expand.

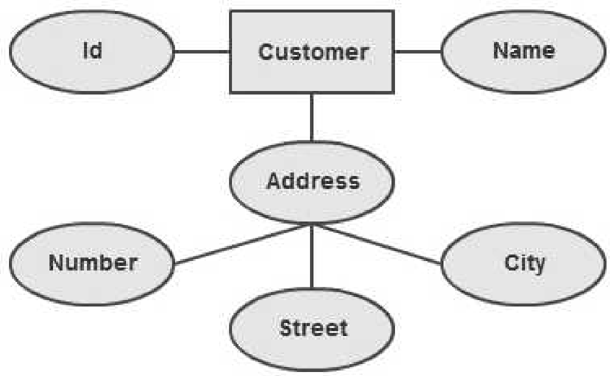


A DATABASE DEVELOPMENT LIFE CYCLE

10(a). Explain E-R Model.

Entity-Relationship Model:
The entity-relationship data model is based on real world objects called entities and relationship among these objects. E-R model is represented graphically by an E-R diagram.
The following are the components of E-R diagram.

1. Rectangle represents Entity set.
2. Ellipses represent attributes.
3. Diamonds represents relationships among entity sets.
4. Lines represent link attributes to entity sets and entity sets to Relationships.



E-R Diagram: An entity-relationship diagram (ERD) is a data modeling technique that creates a graphical representation of the entities, and the relationships between entities, within an information system. Any ER diagram has an equivalent relational table, and any relational table has an equivalent ER diagram.

<u>Entity:</u> The entity is a person, object, place or event for which data is collected. It is equivalent to a database table. An entity can be defined by means of its properties, called attributes. For example, the CUSTOMER entity may have attributes for such things as name, address and telephone number.

<u>Relationship:</u> The relationship is the interaction between the entities.

10(b). Define Normalization. Explain 1NF, 2NF and 3NF.

Normalization: Normalization is a process of evaluating and correcting table structures to minimize data redundancies, thereby reducing the likelihood of data anomalies. The normalization process involves assigning attributes to the tables based on the concept of Relational Data Model.

The objective of normalization is to ensure that each table conforms to the concept of well-formed relations. Normalized tables have the following characteristics.

- ❖ Each table represents a single subject. For example, a course table will contain only data that directly pertains to courses. Similarly, a student table will contain only student data.
- ❖ No data item will be unnecessarily stored in more than one table. The reason for this requirement is to ensure that the data are updated in only one place.
- ❖ All non-prime attribute in a table are dependent on the primary key. The reason for this requirement is to ensure that the data are uniquely identifiable by a primary key value.

❖ Each table is void of insertion, update or deletion anomalies. This is to ensure the integrity and consistency of the data.

1st Normal Form: A table is in first normal form if all the key attributes have been defined and it contains no repeating groups

2nd Normal Form: A table is in second normal form (2NF) if and only if it is in 1NF and every non key attribute is fully functionally dependent on the whole of the primary key (i.e. there are no partial dependencies).

3rd Normal Form:

A relation is said to be in 3NF if
(1). It is in 2NF
(2). There are no transitive dependencies

11(a). Define Table and View. Explain the procedure of creating a table and view in SQL.

Table: Table are defined in three steps.

1. The name of the table is given.
2. Each column is defined, possibly including column constraints.
3. Table constraints are defined.

Create table Tablename(column-name1, data type(number of characters))
(column-name2, data type(number of characters))
……………………………………………………
(column-name, data type(number of characters))

View: A definition of a restricted portion of the database. Views are useful for maintaining confidentiality and restricts access to selected parts of the database and for simplifying frequently used
query types.

The formats of create view command is

Create view V As select statement
As = query specification
V= view name

Create view stud-view as
(Select stud-no, stud-name
From student
Where percentage<=35)

A view is a database object that represents one or more database tables. It doesn't occupy any table Space.

11(b). Explain Client-Server Database Systems.

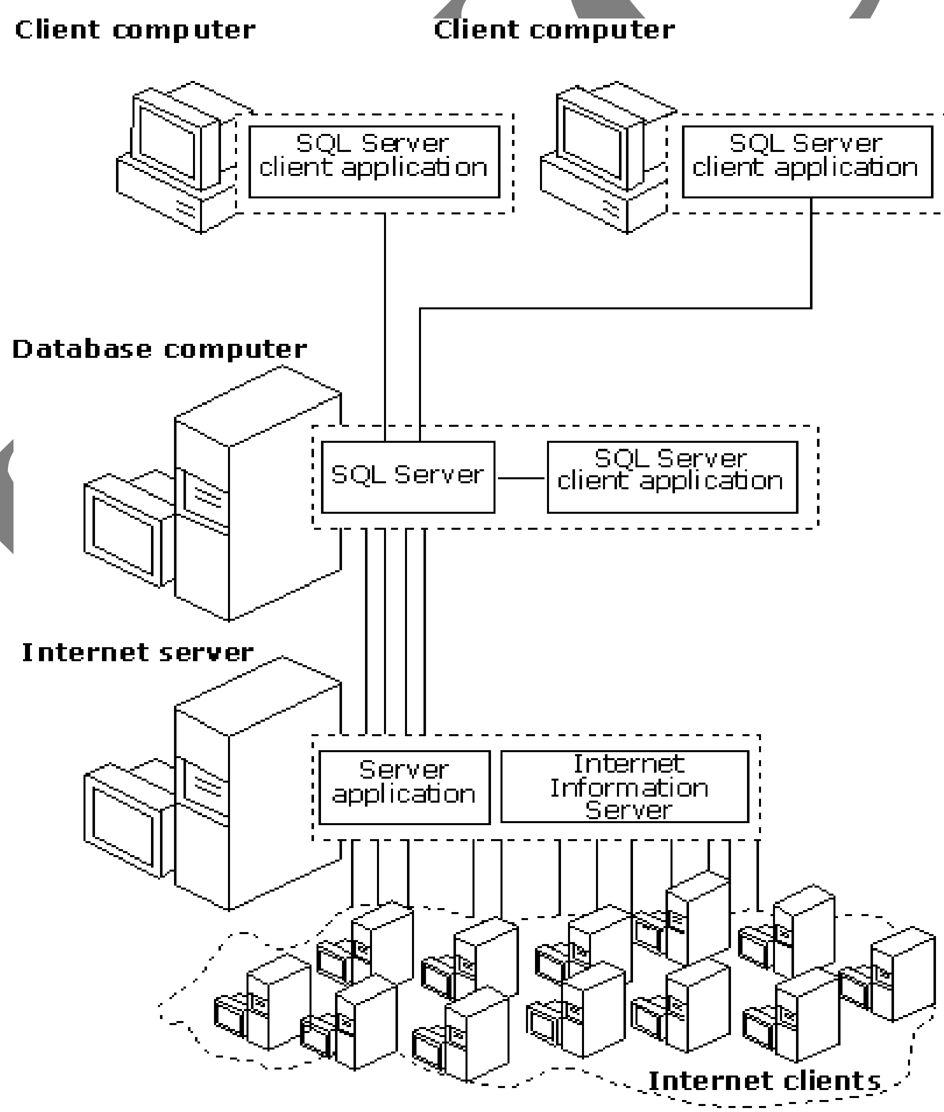
Client/server systems are constructed so that the database can reside on a central computer, known as a server, and be shared among several users. Users access the server through a client or server application:

In a two-tier client/server system, users run an application on their local computer, known as a client, that connects over a network to the server running SQL Server. The client application runs both business logic and the code to display output to the user, and is also known as a thick client.

In a multi tier client/server system, the client application logic is run in two locations:

The thin client is run on the user's local computer and is focused on displaying results to the user.

The business logic is located in server applications running on a server. Thin clients request functions from the server application, which is itself a multithreaded application capable of working with many concurrent users. The server application is the one that opens connections to the database server and can be running on the same server as the database, or it can connect across the network to a separate server operating as a database server.



This is a typical scenario for an Internet application. For example, a server application can run on a Microsoft Internet Information Services (IIS) and service thousands of thin clients running on the Internet or an intranet. The server application uses a pool of connections to communicate with a copy of SQL Server. SQL Server can be installed on the same computer as IIS, or it can be installed on a separate server in the network.

12(a). Define File Organization. Explain the types of File organization.

There are three basic ways of physically organizing files on storage devices.

- Sequential organization
- Indexed-Sequential organization
- Direct organization.

The terms organization and access are often used loosely if not interchangeably. The reason is that the way in which data are stored is closely intertwined with the method access.

1. Sequential File Organization: Sequential file organizational means that records are stored adjacent to one another according to a key such as employee number, account number, and so forth. A conventional implementation arranges the records in ascending order of key values. This is efficient method of organizing records when an application. Such as a payroll program, will be updating a significant number of the stored records.

If a sequential file is maintained on magnetic tape, its records can only be accessed in a sequential manner. That is, if access to the tenth record in sequence is desired generally the preceding nine records must be read. Direct access of a particular record is impossible. Consequently magnetic tapes are not well suited for database operations and are usually relegated log files and recording archival information.

2. Indexed- Sequential File Organization : - When files are sequentially organized on a disk pack, however, direct access of records is possible. Indexed-sequential file organization provides facilities for accessing records both sequentially and directly. Records are stored in the usual physical sequence by primary key. In addition an index of record locations is stored on the disk. This allows records to be accessed sequentially for applications requiring the updating of large numbers of records, as well as providing the ability to access records directly in response to user queries.

3. Direct File Organization: The third type of file organization is called direct or hashed.
It is of two types:

- Static Hash Function
- Dynamic Hash Function

(a). Static Hash Function: The use of hashing is a method of record addressing that eliminates the need for maintaining and searching indexes. The basic idea is that of trading the time and effort associated with storing, maintaining, and searching an index for the time required for the central processing unit (CPU) to execute a hashing algorithm, which generates the record address. The hashing algorithm is a procedure for calculating a record address from some field in the record, usually the key.

(b). Dynamic hash functions: The static hash function is fairly simple. As the database grows, however, the static hash function loses its appeal. One strategy for dealing with this problem is to allocate estimated space for future requirements at the outset; but this wastes storage space. Another scheme is to allocate additional storage and reorganize the file as it grows.

A better approach is provided by the dynamic hash function. This hashing splits and combines blocks as the database grows or shrinks. This ensures efficient space utilization. Moreover, since reorganization involves only one block at a time, the associated overhead is minimal.

Dynamic hashing uses a hash function "h" that has the useful characteristics of randomness and uniformity. It also (typically) uses a 32-bit binary string in order to create and identify block indexes.

12(b). What are the functions and goals of DBA.

Database Administrator is a person with the responsibility of controlling and protecting the data. The DBA should coordinate the design of the database, guide the development and implementation of data security procedures, protect the integrity of data values and make sure system performance is satisfactory.

In a small organization, one person carries out all these responsibilities. Often, these functions are assigned to a group of people. This is most likely in a large organization where DBA responsibilities are divided among several people managed by a chief administrator.

Functions of DBA:

1. Schema definition. The DBA creates the original database schema by executing a set of data definition statements in the DDL.

2. Storage structure and access-method definition: writing a set of definitions translated by the data storage and definition language compiler

3. Schema and physical-organization modification. The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.

4. Granting of authorization for data access. By granting different types of authorization, the database administrator can regulate which parts of the database various users can access. The authorization information is kept in a special system structure that the database system consults whenever someone attempts to access the data in the system.

5. Integrity constraint specification: generating integrity constraints. These are consulted by the database manager module whenever updates occur.

Goals of DBA:

A database must be protected from accidents, such as input or programming errors, from malicious use of the database, and from hardware of software failures that corrupt data. Protection from accidents that cause data inaccuracies is part of the goal of maintaining data integrity. These accidents include failures during transaction processing. Logical errors that violate the assumption that transactions preserve database consistency constraints, and anomalies due to concurrent access to the database (concurrent processing).

Protecting the database from unauthorized or malicious use is termed data security. Although the dividing line between data integrity and data security is not precise, a working definition is as follows:

- Integrity is concerned with making certain that operations performed by users are correct and maintain database consistency.
- Security is concerned with limiting users to performing only those operations that are allowed.
- Having Knowledge of SQL.
- Having Knowledge of Operating System
- Database Initialization: Database initialization is an important part of application deployment. Typically a DBA applies a set of SQL scripts to initialize a database or perform an upgrade.
- Able to create database: The Oracle database administrator must be able to create database objects in its tablespace (either unlimited or with a space quota) with privileges to create session, table, procedure, and views.
- Perform Backup : Backup and recovery procedures are tested regularly to assure their effectiveness in restoring the database after any disruption of service/ a disaster plan has been drawn up and is tested periodically to make sure it works.

The possibility of hardware or software failure requires that database recovery procedures be implemented as well. That is, means must be provided to restore databases that have been corrupted by system malfunctions to a consistent state.

13(a). Explain the need for Distributed Database and Distributed Query Processing.

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:

Need of Distributed Database: -

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.

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 R		*Site 2* 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.

13(b). Explain the functions and capabilities of DBA.

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.