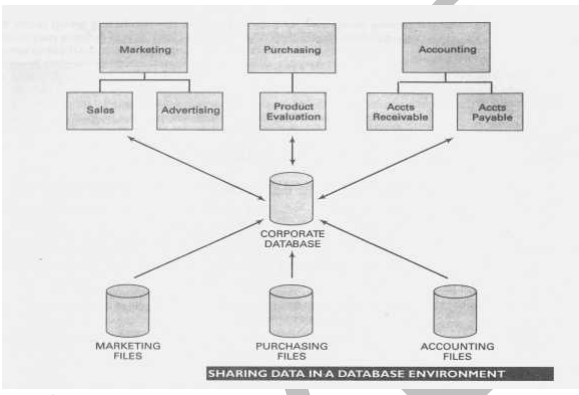
Relational Database Management Systems – April 2010

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

1. Write a short note on Data Sharing.

The data sharing suggests that people in different functional areas use common pool of data, each of their own applications. Without data sharing, the marketing group may have their data files, the purchasing group theirs, the accounting group theirs and so on. Each group benefits only from its own data. The combined data are more valuable than the sum of the data in separate files. Not only does each group continue to have access to its own data but, within reasonable limits of control, they have access to other data as well. The concept of combining data for common use is called data integration.



SHARING DATA IN A DATABASE ENVIRONMENT

2. Explain 3NF with one example.

3NF: A table is in third normal form (3NF) if and only if it is in 2NF and every non key attribute is non transitively dependent on the primary key (i.e. there are no transitive dependencies).

1. Anomalies can occur when a relation contains one or more transitive dependencies.
2. A relation is in 3NF when it is in 2NF and has no transitive dependencies.
3. A relation is in 3NF when 'All non-key attributes are dependent on the key, the whole key and nothing but the key'.

Take the following table structure as an example:

order(order_id, cust, cust_address, cust_contact, order_date, order_total)

Here we should realise that cust_address and cust_contact are functionally dependent on cust which is not a key. To make this table 3NF these attributes must be removed and placed somewhere else.

3. Explain any five aggregate functions of SQL with examples.

Function	Details	Example command
AVG (DISTINCT | ALL) X | indicates OR, Default is ALL.	Finds the average value of field. X. X can be numeric data or character data that	SELECT AVG(basic) FROM salary;

	contains only numeric	
MAX (DISTINCT | ALL) X | indicates OR, Default is ALL	Finds the maximum value in column X. It can be used on any type of data	SELECT MAX (basic) FROM salary;
MIN (DISTINCT | ALL) X | indicates OR, Default is ALL	Finds the minimum value in column X. It can be used for any type of data.	SELECT MIN (basic) FROM salary;
SUM (DISTINCT | ALL) X | indicates OR, Default is ALL	Finds sum of all values of x. It can be used on numeric data	SELECT SUM (basic) FROM salary;
COUNT(*)	Counts the number of rows in the table or the number of rows in a specific group. It includes NULL.	SELECT COUNT(*) FROM salary;
COUNT(Column-name)	Counts the number of non-null values in the column	SELECT COUNT(basic) FROM salary;
COUNT(DISTINCT column Name)	Counts the number of different values in the specified column	SELECT COUNT (DISTINCT basic) FROM salary;

4. Explain the 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 transaction 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:

1. Integrity is concerned with making certain that operations performed by users are correct and maintain database consistency.
2. Security is concerned with limiting users to performing only those operations that are allowed.

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.

5. Explain the advantages of distributed database system.

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.

6. Explain the following disk performance factors: (a) Data Transfer Rate (b) Data Transfer Time.

Data Transfer Rate: - The rate at which data can be read from the disk from the main, memory, or equivalently, the rate at which data are written from main memory to disk. Data Transfer Rate refers to the amount of time required to transfer data from the disk to primary memory. It is a function of rotational speed and the density recorded data. Data Transfer Time is usually expressed in thousands of bytes per second.
Data Transfer Time: - The expected time (T) to access a disk address and transfer a block of data is estimated as

$T = A+R/2+L/2$
Where A is the Access motion time, R is the Rotational delay, L is the length of the block in bytes, and D is the Data Transfer Rate.

7. What is Database Security?

Database integrity problems can be challenging, but they are generally easier to cope with than malicious access to the database, which includes the following:

2. Theft of information
3. Unauthorized modification of data
4. Unauthorized destruction of data

Thus, 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 se 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:

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

Views and security in SQL: -
CREATE VIEW viewname As (select statement)

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. 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 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

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

9 (a). Explain risks and costs involved in database.

Database systems have drawbacks.

The following are the Risks & Costs of a database:

(i). Organizational Conflicts: Pooling data in a common database may not be politically feasible in some organizations. Certain user groups may not be willing to relinquish control over their data to the extent needed to integrate data. Moreover, the risk involved in data sharing – for example, that one group may damage another group's data – and the potential system problems that may limit a group's access to its own data may be viewed as more troublesome than beneficial. Such people problems could prevent the effectual implementation of a database system.

(ii). Development Project Failure: For a variety of reasons, the project to develop a database system may fail. Sometimes management was not fully convinced of the value of the database system in the first place. A database project that seems to be taking too long may be terminated.

A project too large in scope may be almost impossible to complete in a reasonable time. Again, management and users become disenchanted and the project fails.

During the course of a project, key personnel may unexpectedly leave the company. If replacement personnel cannot be found, then the project might not be successfully completed.

(iii). System Failure: When the system goes down, all users directly involved in accessing must wait until the system is functional again. This may require a long wait. Moreover, if the system or application software fails, there may be permanent damage to the database. It is very important, therefore to carefully evaluate all software that will have a direct effect on the database to be certain that it is as free as errors as possible. If the organization does not use a database, it is not exposed to this risk, since the data and its software are distributed.

(iv). Overhead Costs: The database approach may require an investment in both hardware and software. The hardware to run large DBMS must be efficient and will generally require more main memory and disk storage than simpler file-based system. Tape drivers for rapidly backing up the database are also required. In addition, the DBMS itself may be quite expensive.

The DBMS may also need increase operating costs, since it requires more execution time. For example, an application system using a DBMS will usually execute more slowly than a system not using a DBMS.

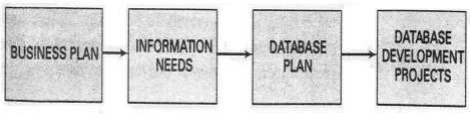
(v). Need for Sophisticated Personnel: The database administration function requires skilled personnel who are capable of coordinating the needs of different user groups, designing views, integrating those views into a single schema, establishing data recovery procedures and fine tuning the physical structure of the database to meet acceptable performance criteria. There is a risk involved in identification of personnel for the DBA, since if no person having the requisite skills can be found, the DBA function may not be properly performed. This could result in significant problems and may even result in the failure of a database implementation.

9(b). Explain about the strategic database planning.

Database Planning is a strategic corporate effort to determine information needs for an extended period. A successful database planning project will precede operational projects to design and implement new databases to satisfy the organization's information needs.

The Need for Database Planning: Database planning has significant advantages:

- It expresses management's current understanding of the information resource.
- It identifies and justifies resource requirements.
- It identifies opportunities for effective resource management including collaboration among departments or divisions within the organizations.
- It specifies action pans for achieving objectives.
- It can provide a powerful stimulus and sense of direction to employees at all levels, focusing their efforts, increasing their productivity and making them feel that they are a genuine part of the enterprise.



The Database Planning Project: Strategic Database Planning is initiated by senior management. They allocate resources and identify personnel to participate in the project. With their commission from management, team members have resources needed to carry out a successful project.

The project team should be extensive experience in information systems and other functional areas of the company. A group of four full-time members, two from information systems and two acquainted with most other areas of the company. All team members should be skilled and respected employees, since their work will have a major impact on the organization for many years. If they are not skilled in a methodology for carrying out the study, an outside consultant should be employed as an advisor to train the team in a suitable methodology. The project team leader should be a consultant but a permanent employee and possibly the head of the database administration.

During the project, the team interacts with senior managers from all the primary user areas. The senior end users identify the principal processes, activities, and entities used in manual or automated information processing. The project team synthesizes these data into a corporate information model included as part of the comprehensive database plan.

A report covering at least the next five should be delivered to senior management. This report will include analyses of the following:

- Information needs of the functional areas.
- Information needs of different management levels.
- Information needs of the geographical locations.
- A model of this information needs.
- Anticipated data volumes by geographical location projects for the period under study.
- A preliminary estimate of costs associated with system upgrades.
- Recommendations for detailed development of new or enhanced databases with schedules.

10(a). State all the normal terms. Explain 3NF with example.

Normalization is the process of converting the table into standard form.

It is of different types:

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).

Boyce-Codd Normal Form: A table is in Boyce-Codd normal form (BCNF) if and only if it is in 3NF and every determinant is a candidate key.

3rd Normal Form: A table is in third normal form (3NF) if and only if it is in 2NF and every non key attribute is non transitively dependent on the primary key (i.e. there are no transitive dependencies)

4. Anomalies can occur when a relation contains one or more transitive dependencies.
5. A relation is in 3NF when it is in 2NF and has no transitive dependencies.
6. A relation is in 3NF when 'All non-key attributes are dependent on the key, the whole key and nothing but the key'.

Take the following table structure as an example:

order(<u>order_id</u>, cust, cust_address, cust_contact, order_date, order_total)

Here we should realise that cust_address and cust_contact are functionally dependent on cust which is not a key. To make this table 3NF these attributes must be removed and placed somewhere else.

You must also note the use of calculated or derived fields. Take the example where a table contains PRICE, QUANTITY and EXTENDED_PRICE where EXTENDED_PRICE is calculated as QUANTITY multiplied by PRICE. As one of these values can be calculated from the other two then it need not be held in the database table. Do not assume that it is safe to drop any one of the three fields as a difference in the number of decimal places between the various fields could lead to different results due to rounding errors. For example, take the following fields:

- AMOUNT - a monetary value in home currency, to 2 decimal places.
- EXCH_RATE - exchange rate, to 9 decimal places.
- CURRENCY_AMOUNT - amount expressed in foreign currency, calculated as AMOUNT multiplied by EXCH_RATE.

If you were to drop EXCH_RATE could it be calculated back to its original 9 decimal places?

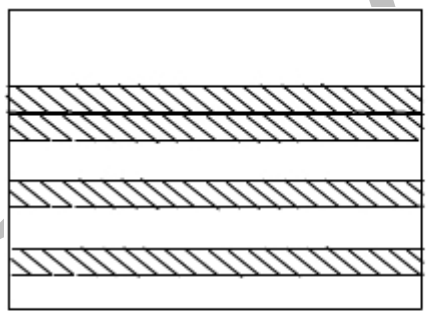
Reaching 3NF is adequate for most practical needs, but there may be circumstances which would benefit from further normalization.

10(b). What is relational algebra? Explain the operators in relational algebra.

Relational Algebra: It is a collection of operators that are used to manipulate entire relations. These operators are used to select tuples or to combine tuples from individual relations for specifying a query on the database.

The eight relational algebra operators are

1. SELECT To retrieve specific tuples/rows from a relation.



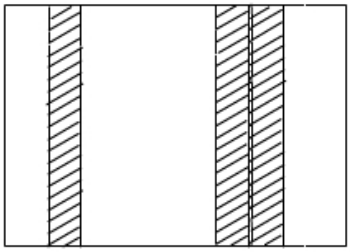
SELECT

SELECT orders placed by customer number 002

$$\sigma_{Cust\#=002}(ORD\_AUG)$$

Ord #	Ord Date	Cust #
101	02-08-94	002
104	18-09-94	002

2. PROJECT To retrieve specific attributes/columns from a relation.



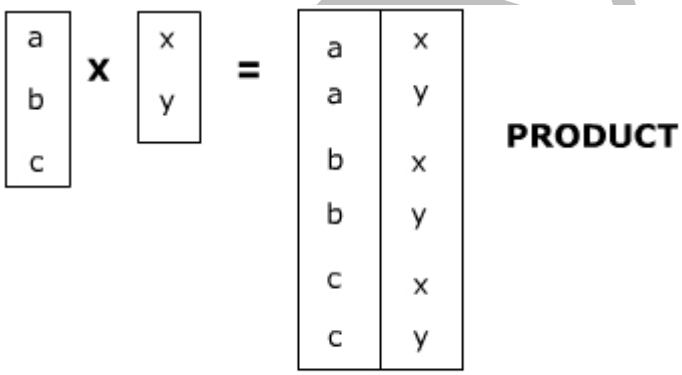
Project

PROJECT description and price from Items relation

$$\Pi_{Descr,\ Price}(ITEMS)$$

Descr	Price
Power Supply	4000
101-Keyboard	2000
Mouse	800
MS-DOS 6.0	5000
MS-Word 6.0	8000

3. PRODUCT To obtain all possible combination of tuples from two relations.



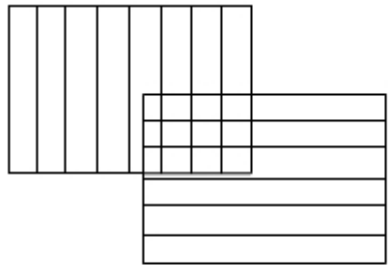
PRODUCT

ORD_AUG X CUSTOMERS

Ord#	OrdDate	O.Cust#	C.Cust#	CustName	City
101	02-08-94	002	001	Shah	Bombay
101	02-08-94	002	002	Srinivasan	Madras
101	02-08-94	002	003	Gupta	Delhi
101	02-08-94	002	004	Banerjee	Calcutta

101	02-08-94	002	005	Apte	Bombay
102	11-08-94	003	001	Shah	Bombay
102	11-08-94	003	002	Srinivasan	Madras

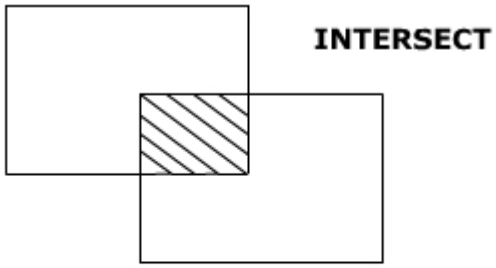
4. UNION To retrieve tuples appearing in either or both the relations participating in the UNION.



ORD_JUL ∪ ORD_AUG

Ord #	Ord Date	Cust #
101	03-07-94	001
102	27-07-94	003
101	02-08-94	002
102	11-08-94	003
103	21-08-94	003
104	28-08-94	002
105	30-08-94	005

5. INTERSECT- To retrieve tuples appearing in both the relations participating in the INTERSECT.



Customers who have placed orders in July and August

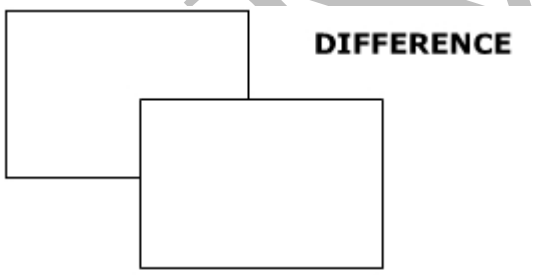
$$\prod_{Cust\#}(ORD\_JUL) \cap \prod_{Cust\#}(ORD\_AUG)$$

Eg:
To retrieve Cust# of Customers who've placed orders in July and in August

Cust#

003

6. DIFFERENCE To retrieve tuples appearing in the first relation participating in the DIFFERENCE but not the second.



Customers who have placed orders in July
BUT not in August

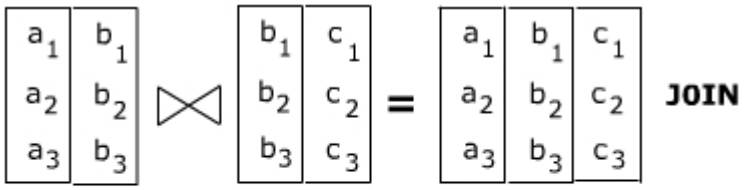
$$\prod_{Cust\#}(ORD\_JUL) - \prod_{Cust\#}(ORD\_AUG)$$

Eg:
To retrieve Cust# of Customers who've placed orders in July but not in August

Cust#

001

7. JOIN To retrieve combinations of tuples in two relations based on a common field in both the relations.



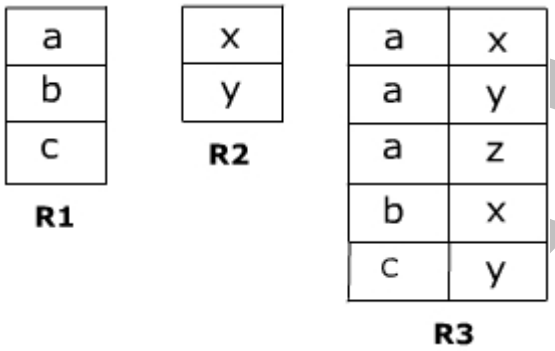
Join of ORD_AUG with CUSTOMERS

ORD_AUG ⋈ CUSTOMERS

Eg:

ORD_AUG join CUSTOMERS (here, the common column is Cust#)

8. DIVIDE Consider the following three relations:



a
b
c

R1

x
y

R2

a	x
a	y
a	z
b	x
c	y

R3

R1 divide by R2 per R3 gives:

a

Thus the result contains those values from R1 whose corresponding R2 values in R3 include all R2 values.

11(a). Define a view with example. Explain different operators on a view.

Views are mechanism of data independence. They provide automatic security of hidden data. A view is window to a specific use of the database.

Usage of views:

- Views are used to secure attributes that contain private information.

- Views can be used for creating short cuts for complex queries involving multiple tables.
- Views can be created with a Check option that prevents the updating of rows and attributes that are not part of the view but are part of complete database

Creating a View

For creating a view following command is used:

Syntax: CREATE or REPLACE VIEW <view-name> AS <query>;

Example: CREATE VIEW emp_view AS
SELECT e_code, e_name FROM employee;

Please note that while defining a view an ORDER BY clause cannot be used.

Displaying the view

For displaying all the attributes of the View give the following command:
SELECT * FROM emp_view;

For displaying the attributes of the view use the following command:
DESC emp_view;

Deleting a View

For deleting a View please use the following command:
DROP VIEW emp_view;

Updating through a View

The data through the views cannot be updated if it include JOIN, SET operations, GROUP BY, DISTINCT, AGGREGATE function; or have composite attributes; or does not contain the field of the base table that contains NOT NULL including primary key; or violates WITH CHECK OPTION clause.

11(b). Explain the various DML commands in SQL with examples.

INSERT Statement

To get data into a database, we need to use the 'insert' statement. The general syntax is:

```
INSERT INTO <table-name> (<column1>,<column2>,<column3>,...)
    VALUES (<column-value1>,<column-value2>,<column-value3>);
```

The column names (i.e.: `column1`, etc.) must correspond to column values (i.e.: `column-value1`, etc.). There is a short-hand for the statement:

```
INSERT INTO <table-name>
    VALUES (<column-value1>,<column-value2>,<column-value3>);
```

In which the column values must correspond exactly to the order columns appear in the 'create table' declaration. It must be noted, that this sort of statement should (or rather, *must*) be avoided! If someone changes the table, moves columns around in the table declaration, the code using the shorthand insert statement will fail.

A typical example, of inserting the 'person' record we've created earlier would be:

```
INSERT INTO PERSON(PERSONID,LNAME,FNAME,DOB)
    VALUES(1,'DOE','JOHN','1956-11-23');
```

SELECT Statement

Probably the most used statement in all of SQL is the `SELECT` statement. The select statement has the general format of:

```
SELECT <column-list>
FROM <table-list>
WHERE <search-condition>
```

UPDATE Statement

The update statement is used for changing records. The general syntax is:

```
UPDATE <table-name>
SET <column1> = <value1>, <column2> = <value2>, ...
WHERE <criteria>
```

The criteria is what selects the records for update. The 'set' portion indicates which columns should be updated and to what values. An example of the use would be:

```
UPDATE PERSON
SET FNAME='Clark', LNAME='Kent'
WHERE FNAME='Superman';
```

DELETE Statement

The 'delete' is used to remove elements from the database. The syntax is very similar to update and select statements:

```
DELETE FROM <table-name>
WHERE <criteria>
```

Basically we select which records we want to delete using the where clause. An example use would be:

```
DELETE FROM PERSON
WHERE PERSONID=12345;
```

12(a). What is file organization? Explain the different types of file organization.

File organization is the method in which the way records are stored in the file.

There are three basic ways of physically organizing files on storage devices. Sequential organization, indexed-sequential organization and direct organization.

This is not an entire set of all organization options available but those that are omitted are modifications of these basic organization types. Therefore it is not necessary to be exhaustive in order to cover the essential concepts.

In discussing the topic at hand, 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.

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 acesending 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.

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.

Direct File Organization: - We have studied two forms of file organization: Sequential and indexed sequential. We have concurrently outlined the two associated methods of file access: sequential access and direct access. Records in a simple sequential file organization can be accessed directly, as well as sequentially. We now turn to a discussion of a third type of file organization called direct or hashed. Only direct access methods are applicable to this type of file organization.

12(b). Explain the mapping logical data structures and implementing a DBMS.

A record is a collection of data items and is the unit for data storage at the logical or file level. The application program usually reads a complete record from the database. A record may consist of different fields and each field corresponds to an attribute of the record.

The records may of fixed size or may be variable in length; one block may contain multiple records. When the records are called unspanned , Whereas for spanned records, portions of a single record may lie in different blocks.

Different methods of arranging records in blocks are called blocking of records.

1. Fixed blocking for fixed-length records.
2. Variable length blocking for unspanned variable length records
3. Variable length blocking for spanned records.

Magnetic tape files allow only sequential file organization where the records are stored in a sequential order only. Whereas with magnetic disc, we can have 3 types

of file organization. In indexed file organization ,the records are stored on index basis. In relative file organization, the records are stored on relative key which is evaluated by hashing method.

In logical organization of data refers to different formats of data structure, whereas in physical organization is in the form of storage structure. Storage is represented through files and collection of files is known as Directory.

13(a). What are the factors involved in selecting and implementing a DBMS.

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.

13(b). Explain about distributed query processing.

B) 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.