a) What are the responsibilities of Database Administrator?
Database administrators (DBAs) use specialized software to store and organize data. The role may include capacity planning, installation, configuration, database design, migration, performance monitoring, security, troubleshooting, as well as backup and data recovery .

b) Define primary key
A primary key is a key in a relational database that is unique for each record. It is a unique identifier, such as a driver license number, telephone number (including area code), or vehicle identification number .A relational database must always have one and only one primary key.

c) What is canned transaction?
Canned transactions are standard types of queries and updates which are frequently used by Naive end users to constantly querying and updating database.

d) Write the Relational Algebra query to select the tuples of all employees who work in department 4 and make over $25,000 per year

e) List out the DDL commands in SQL.
A DDL is a language used to define data structures and modify data. For example, DDL commands can be used to add, remove, or modify tables within in a database. DDL statements create, modify, and remove database objects such as tables, indexes, and users. Common DDL statements are CREATE, ALTER, and DROP.

f) Give the syntax for INSERT command in SQL.
INSERT INTO TABLE_NAME (column1, column2, column3,...columnN)
VALUES (value1, value2, value3,...valueN);

g) What is a primary index?
Primary index is defined on an ordered data file. The data file is ordered on a **key field**.
The key field is generally the primary key of the relation.

h) When a relation is said to be in 1NF?
A relation is in first normal form if and only if the domain of each attribute contains only atomic (indivisible) values, and the value of each attribute contains only a single value from that domain.

i) What is a join dependency? Give an example.
A join dependency (JD), denoted by JD(R1, R2, ..., Rn), specified on relation schema R, specifies a constraint on the states r of R. The constraint states that every legal state r of R should have a nonadditive join decomposition into R1, R2, ..., Rn.

j) What is Conflict Serializability?
A schedule is called conflict serializable if it can be transformed into a serial schedule by swapping non-conflicting operations.

k) What are the two pitfalls of lock-based protocols?
1.Lock-based protocols may cause deadlocks/starvation if not implemented carefully.
2.The tradeoff between the locking granularity and locking overhead is difficult to calculate and generally depends on the application workload pattern.

l) What is a Thomas Write rule?

Thomas Write Rule provides the guarantee of serializability order for the protocol. It improves the Basic Timestamp Ordering Algorithm. The basic Thomas write rules are as follows:

If TS(T) < R_TS(X) then transaction T is aborted and rolled back, and operation is rejected.

If TS(T) < W_TS(X) then don't execute the W_item(X) operation of the transaction and continue processing.

If neither condition 1 nor condition 2 occurs, then allowed to execute the WRITE operation by transaction Ti and set W_TS(X) to TS(T).

UNIT-I

2 a) List and explain the different applications that use DBMSs?

Listing any 3 applications---2M
Explaination of any 3 applications---4M

Railway Reservation system
Library Management System
Banking
Universities and colleges
Social Media sites
Telecommunication

b) Describe Centralized DBMSs Architecture and Two-Tier Client/Server Architecture for DBMSs with neat sketch.

Centralized DBMSs Architecture ---3M

Earlier architectures used mainframe computers to provide the main processing for all system functions, including user application programs and user interface programs, as well as all the DBMS functionality. The reason was that most users accessed such systems via computer terminals that did not have processing power and only provided display capabilities. Therefore, all processing was performed remotely on the computer system, and only display information and controls were sent from the computer to the display terminals, which were connected to the central computer via various types of communications networks.

At first, database systems used these computers similarly to how they had used display terminals, so that the DBMS itself was still a centralized DBMS in which all the DBMS functionality, application program execution, and user inter-face processing were carried out on one machine. Figure illustrates the physical components in a centralized architecture. Gradually, DBMS systems started to exploit the available processing power at the user side, which led to client/server DBMS architectures.

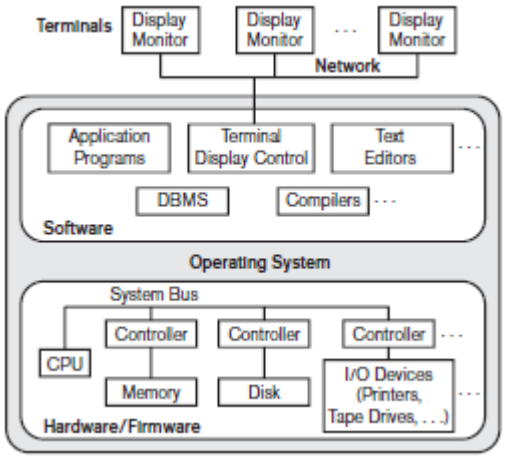


Fig:Centralized Architecture

Client/Server Architectures-----3M

The client/server architecture was developed to deal with computing environments in which a large number of PCs, workstations, file servers, printers, database servers, Web servers, e-mail servers, and other software and equipment are connected via a network. The idea is to define specialized servers with specific functionalities. For example, it is possible to connect a number of PCs or small workstations as clients to a file server that maintains the files of the client machines. Another machine can be designated as a printer server by being connected to various printers; all print requests by the clients are forwarded to this machine. Web servers or e-mail servers also fall into the specialized server category. The resources provided by specialized servers can be accessed by many client machines. Theclient machines provide the user with the appropriate interfaces to utilize these servers, as well as with local processing power to run local applications.

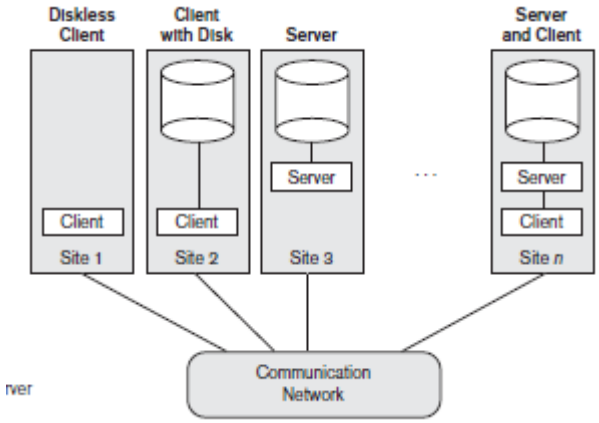


Fig: Two-Tier Client server Architecture

OR

3 a) Explain about component modules of a DBMS and their interactions with neat sketch. 6M

DBMS Component Modules Explanation--4M

The figure is divided into two halve. The top half of the diagram refers to the various users of the database environment and their interfaces. The lower half demonstrates the internals of the DBMS responsible for storage of data and processing of transaction.

The database and the DBMS catalogue are usually stored on disk. Access to the disk is principally controlled by operating system (OS). This includes disk input/Output. A higher point stored data manager module of DBMS controls access to DBMS information that is stored on the disk.

If we consider the top half of the figure it shows interface to casual users, DBA staff, application programmers and parametric users.

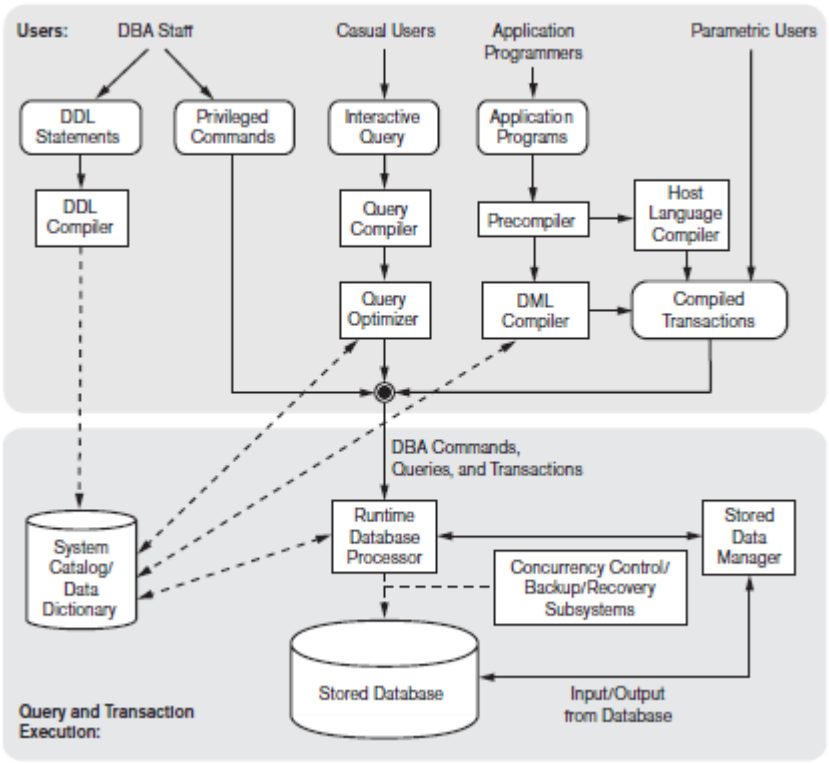
The DDL compiler specified in the DDL, processes schema definitions as well as stores the description of the schema in the DBMS Catalogue. The catalogue includes information such as names and sizes of the sizes of the files and data types of data of data items.

Storage particulars of every file mapping information among schemas as well as constraints.

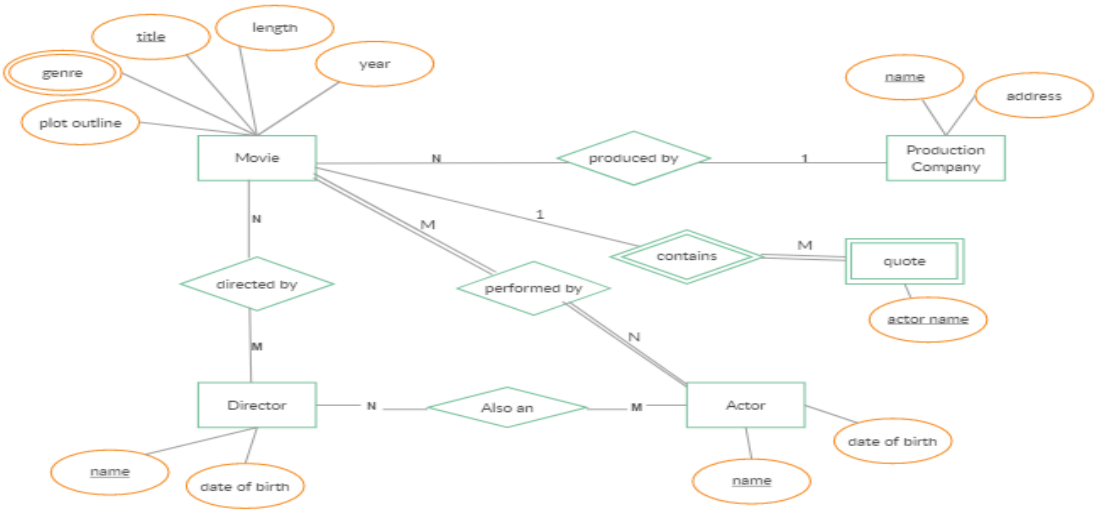
Casual users as well as persons with occasional need of information from database interact using some of interface which is interactive query interface. The queries are parsed analyse for correctness of the operations for the model. The names of the data elements as well as therefore on by a query compiler that compiles them into internal form. The interior query is subjected to query optimization. The query optimizer is worried with rearrangement and possible recording of operations and eliminations of redundancies.

Application programmer inscribes programs in host languages. The precompiled take out DML commands from an application program.

Diagram--2M



b) Construct ER Diagram for a MOVIES database schema. Identify entities, relationships and weak entity sets if any



UNIT-II

4 a) Write short notes on Tuple and Domain Relational Calculus.

[Tuple relational calculus Explanation--2M]
[Example--1M]

The tuple relational calculus is based on specifying a number of tuple variables. Each tuple variable usually ranges over a particular database relation, meaning that the variable may take as its value any individual tuple from that relation. A simple tuple relational calculus query is of the form:

{t | COND(t)}

where t is a tuple variable and COND(t) is a conditional (Boolean) expression involving t that evaluates to either TRUE or FALSE for different assignments of tuples to the variable t. The result of such a query is the set of all tuples t that evaluate COND(t) to TRUE. These tuples are said to satisfy COND(t). For example, to find all employees whose salary is above $50,000, we can write the following tuple calculus expression:

{t | EMPLOYEE(t) AND t.Salary>50000}

The condition EMPLOYEE(t) specifies that the range relation of tuple variable t is EMPLOYEE. Each EMPLOYEE tuple t that satisfies the condition t.Salary>50000 will be retrieved.

The above query retrieves all attribute values for each selected EMPLOYEE tuple t. To retrieve only some of the attributes—say, the first and last names—we write

{t.Fname, t.Lname | EMPLOYEE(t) AND t.Salary>50000}

Informally, we need to specify the following information in a tuple relational calculus expression:

■ For each tuple variable t, the range relation R of t. This value is specified by a condition of the form R(t). If we do not specify a range relation, then the variable t will range over all possible tuples "in the universe" as it is not restricted to any one relation.

■ A condition to select particular combinations of tuples. As tuple variables range over their respective range relations, the condition is evaluated for every possible combination of tuples to identify the selected combinations for which the condition evaluates to TRUE.

■ A set of attributes to be retrieved, the requested attributes. The values of these attributes are retrieved for each selected combination of tuples.

[Domain relational calculus Explanation--2M]
[Example--1M]

There is another type of relational calculus called the domain relational calculus, or simply, domain calculus. Domain calculus differs from tuple calculus in the type of variables used in formulas: Rather than having variables range over tuples, the variables range over single values from domains of attributes. To form a relation of degree n for a query result, we must have n of these domain variables—one for each attribute. An expression of the domain calculus is of the form {x1, x2, ..., xn | COND(x1, x2, ..., xn, xn+1, xn+2, ..., xn+m)}

where x1, x2, ..., xn, xn+1, xn+2, ..., xn+m are domain variables that range over domains (of attributes), and COND is a condition or formula of the domain relational calculus.

A formula is made up of atoms. The atoms of a formula are slightly different from those for the tuple calculus and can be one of the following:

1. An atom of the form R(x1, x2, ..., xj), where R is the name of a relation of degree j and each xi, $1 \leq i \leq j$, is a domain variable. This atom states that a list of values of <x1, x2, ..., xj> must be a tuple in the relation whose name is R, where xi is the value of the ith attribute value of the tuple. To make a domain calculus expression more concise, we can drop the commas in a list of variables; thus, we can write:

{x1, x2, ..., xn | R(x1 x2 x3) AND ...} instead of:

{x1, x2, ... , xn | R(x1, x2, x3) AND ...}

2. An atom of the form xi op xj, where op is one of the comparison operators in the set {=, <, ≤, >, ≥, ≠}, and xi and xj are domain variables.

3. An atom of the form xi op c or c op xj, where op is one of the comparison operators in the set {=, <, ≤, >, ≥, ≠}, xi and xj are domain variables, and c is a constant value.

b) Consider the following relational database:

employee(e-name, street, city)
works(e-name, c-name, salary)
company(c-name, city)
manages(e-name, m-name)

Give an expression to find the names and cities of residence of all employees who work for the First Bank Corporation, in

i) the relational algebra, ii) the tuple relational calculus, iii) the domain relational calculus.

OR

5 a) Consider the following tables:

WORKS(Pname,Cname,City)
LIVES(Pname,Street,City)
LOCATED_IN(Cname,City)
MANAGER(Pname,Mgename) where Pname=Person name,Cname=Company name and Mgrname= Manager name.

Write the SQL for the following:

i) List the names of the people who work for the company Wipro along with the cities they live in.
ii) Find the names of the persons who live and work in the same city.
iii) Find the names of the persons who do not work for 'Infosys'.

b) What is a view? How can it be created? Explain with an example? 6M

Definition--2M

A view in SQL terminology is a single table that is derived from other tables.These other tables can be base tables or previously defined views. A view does not necessarily exist in physical form; it is considered to be a virtual table, in contrast to base tables, whose tuples are always physically stored in the database.

Syntax---2M

CREATE VIEW view_name AS SELECT column1, column2..... FROM table_name WHERE condition;

Explanation with any example----2M

UNIT-II

6 a)What is B^+-Tree? Explain in detail with suitable example? 6M

B+ Tree

A B+ tree is a balanced binary search tree that follows a multi-level index format. The leaf nodes of a B+ tree denote actual data pointers. B+ tree ensures that all leaf nodes remain at the same height, thus balanced. Additionally, the leaf nodes are linked using a link list; therefore, a B+ tree can support random access as well as sequential access.

Structure of B+ Tree

Every leaf node is at equal distance from the root node. A B+ tree is of the order **n** where **n** is fixed for every B+ tree.

Internal nodes

- Internal (non-leaf) nodes contain at least ⌈n/2⌉ pointers, except the root node.
- At most, an internal node can contain **n** pointers.

Leaf nodes

- Leaf nodes contain at least ⌈n/2⌉ record pointers and ⌈n/2⌉ key values.
- At most, a leaf node can contain **n** record pointers and **n** key values.
- Every leaf node contains one block pointer **P** to point to next leaf node and forms a linked list.

Insert algorithm for B+ Tree

Leaf Page Full	Index Page FULL	Action
NO	NO	Place the record in sorted position in the appropriate leaf page
YES	NO	1. Split the leaf page 2. Place Middle Key in the index page in sorted order. 3. Left leaf page contains records with keys below the middle key. 4. Right leaf page contains records with keys equal to or greater than the middle key.
YES	YES	1. Split the leaf page. 2. Records with keys < middle key go to the left leaf page. 3. Records with keys >= middle key go to the right leaf page. 4. Split the index page. 5. Keys < middle key go to the left index page. 6. Keys > middle key go to the right index page. 7. The middle key goes to the next (higher level) index. IF the next level index page is full, continue splitting the index pages.

Deletion algorithm for B+ tree

Leaf Page Below Fill Factor	Index Page Below Fill Factor	Action
NO	NO	Delete the record from the leaf page. Arrange keys in ascending order to fill void. If the key of the deleted record appears in the index page, use the next key to replace it.
YES	NO	Combine the leaf page and its sibling. Change the index page to reflect the change.
YES	YES	1. Combine the leaf page and its sibling. 2. Adjust the index page to reflect the change. 3. Combine the index page with its sibling. Continue combining index pages until you reach a page with the correct fill factor or you reach the root page.

b) Compare BCNF with 4NF. Quote examples 6M

Boyce-Codd Normal Form (BCNF) BCNF--2M

BCNF stands for "*Boyce-Codd Normal Form*". This normal form also known as the 3.5 Normal form of database normalization. To achieve BCNF, the database must be already achieved to third normal form. Then following steps should be done to achieve the BCNF.

1. Identify all candidate keys in the relations
2. Identify all functional dependencies in the relations.
3. If there are functional dependencies in the relation, where their determinants are not candidate keys for the relation, remove the functional dependencies by placing them in a new relation along with a copy of their determinant.

Fourth Normal Form 4NF--2M

Database must be in third normal form, before normalizing it to the fourth normal form. If the database is already in third normal form, next step should be to remove the multi-valued dependencies. (If one or more rows imply the presence of one or more other rows in the same table, it is called multi-valued dependency.)

Difference between BCNF and 4NF (Fourth Normal Form) Comparing---2M

• Database must be already achieved to 3NF to take it to BCNF, but database must be in 3NF and BCNF, to reach 4NF.

• In fourth normal form, there are no multi-valued dependencies of the tables, but in BCNF, there can be multi-valued dependency data in the tables.

OR

7 a)What is functional dependency? Explain its usage in database design. 6M

Definition---3M

A functional dependency, denoted by X → Y, between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuples that can form a relation state r of R. The constraint is that, for any two tuples t1 and t2 in r that have t1[X] = t2[X], they must also have t1[Y] = t2[Y]. This means that the values of the Y component of a tuple in r depend on, or are
determined by, the values of the X component. The abbreviation for functional dependency is FD or f.d. The set of attributes X is called the left-hand side of the FD, and Y is called the right-hand side.

Usage---3M

Functional Dependency avoids data redundancy. Therefore same data do not repeat at multiple locations in that database
It helps you to maintain the quality of data in the database
It helps you to defined meanings and constraints of databases
It helps you to identify bad designs
It helps you to find the facts regarding the database design

b) List and Discuss the inference rules for functional dependencies. Give relevant examples--6M

Reflexive rule: If X is a set of attributes and Y is_subset_of X, then X holds a value of Y.
Augmentation rule: When x -> y holds, and c is attribute set, then ac -> bc also holds. That is adding attributes which do not change the basic dependencies.
Transitivity rule: This rule is very much similar to the transitive rule in algebra if x -> y holds and y -> z holds, then x -> z also holds. X -> y is called as functionally that determines y.

UNIT-IV

8 a) What is a transaction? Explain ACID properties of a transaction. 6M

A transaction is an atomic unit of work that should either be completed in its entirety or not done at all. **Transaction Definition--2M**

Transactions should possess several properties, often called the ACID properties; they should be enforced by the concurrency control and recovery methods of the DBMS. The following are the ACID properties: **ACID Properties--4M**

- **Atomicity.** A transaction is an atomic unit of processing; it should either be performed in its entirety or not performed at all.
- **Consistency preservation.** A transaction should be consistency preserving, meaning that if it is completely executed from beginning to end without interference from other transactions, it should take the database from one consistent state to another.
- **Isolation.** A transaction should appear as though it is being executed in isolation from other transactions, even though many transactions are executing concurrently. That is, the execution of a transaction should not be interfered with by any other transactions executing concurrently.
- **Durability or permanency.** The changes applied to the database by a committed transaction must persist in the database. These changes must not be lost because of any failure.

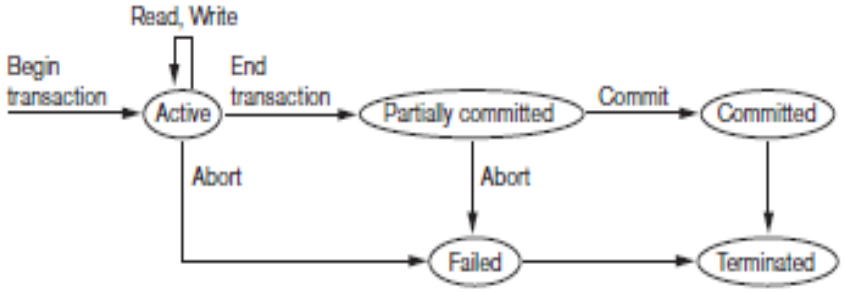
b) Draw and explain the life cycle of transaction. 6M

Explanation--4M

A transaction goes into an **active state** immediately after it starts execution, where it can execute its READ and WRITE operations. When the transaction ends, it moves to the **partially committed state**. At this point, some recovery protocols need to ensure that a system failure will not result in an inability to record the changes of the transaction permanently Once this check is successful, the transaction is said to have reached its commit point and enters the **committed state**. When a transaction is committed, it has concluded its execution successfully and all its changes must be recorded

permanently in the database, even if a system failure occurs. However, a transaction can go to the **failed state** if one of the checks fails or if the transaction is aborted during its active state. The transaction may then have to be rolled back to undo the effect of its WRITE operations on the database. The **terminated state** corresponds to the transaction leaving the system.

Transaction state diagram----2M



OR

9 a) Explain different types of transaction failures in DBMS. 6M

Typesof failures that causes a transaction to fail

A transaction may fail due to one or more following reasons:

Computer failure (System crash) : A hardware, software, or network error may cause a computer to fail or hang during an ongoing transaction.

Transaction error: This error happens when user aborts the transaction, or logical programming error, or some operations like division by zero, etc.

External causes, exception conditions : A transaction may be failed if necessary data not found, or an exception condition met. For example, customer A tries to purchase an item online. Assume that only two such items are available when A starts his transaction and both of them purchased by other customers. When A reaches end of the transaction, the items are not available hence the transaction is cancelled or failed.

Concurrency control enforcement: If a transaction causes deadlock or violates serializability, then concurrency control schemes will force the transaction to abort.

Disk failure: Some disk blocks may lose their data because of a read or write malfunction or because of a disk read/write head crash. This may happen during a read or a write operation of the transaction.

b) Describe the role of checkpoints in database recovery mechanism 6M

Explanation of Any Relevant points can be considerd---6M

Checkpoint is a mechanism where all the previous logs are removed from the system and stored permanently in a storage disk. Checkpoint declares a point before which the DBMS was in consistent state, and all the transactions were committed

The recovery manager of a DBMS must decide at what intervals to take a checkpoint.

The interval may be measured in time say, every m minutes or in the number t of committed transactions since the last checkpoint,where the values of m or t are system parameters. Taking a checkpoint consists of the following actions:

1. Suspend execution of transactions temporarily.
2. Force-write all main memory buffers that have been modified to disk.
3. Write a [checkpoint] record to the log, and force-write the log to disk.
4. Resume executing transactions..

Scheme prepared by **Signature of HOD, IT Dept**

Paper Evaluators:

Sno	Name of the college	Nameof the Examiner	Signature