Tulsiramji Gaikwad-Patil College Engineering and Technology
Department of Computer Science and Engineering
Semester: B.E. V Semester
Subject: Database Management System
Solution Set: Summer 2018

Q1 a) Explain different types of data models used in database.

Ans-

A Database model defines the logical design and structure of a database and defines how data will be stored, accessed and updated in a database management system. While the **Relational Model** is the most widely used database model, there are other models too:

- Hierarchical Model
- Network Model
- Entity-relationship Model
- Relational Model

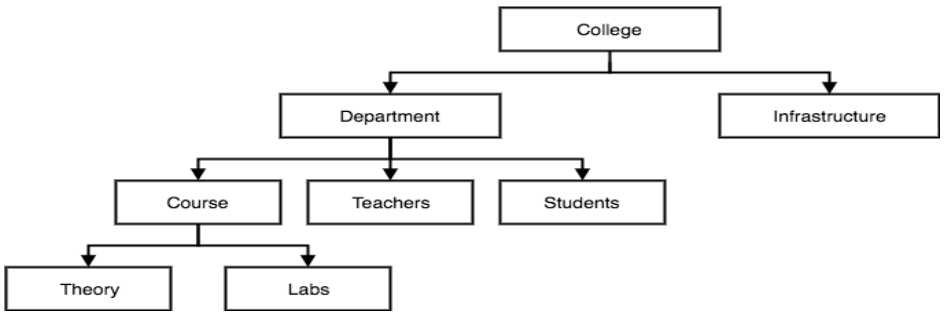
Hierarchical Model

This database model organises data into a tree-like-structure, with a single root, to which all the other data is linked. The heirarchy starts from the **Root** data, and expands like a tree, adding child nodes to the parent nodes.

In this model, a child node will only have a single parent node.

This model efficiently describes many real-world relationships like index of a book, recipes etc.

In hierarchical model, data is organised into tree-like structure with one one-to-many relationship between two different types of data, for example, one department can have many courses, many professors and of-course many students.

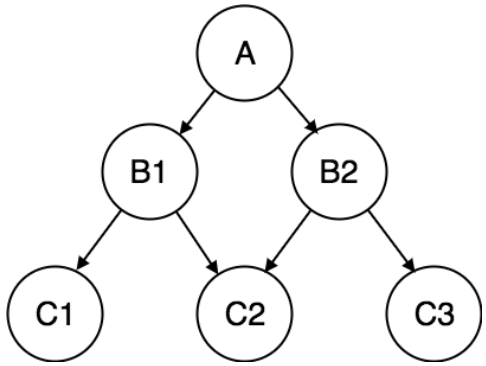


Network Model

This is an extension of the Hierarchical model. In this model data is organised more like a graph, and are allowed to have more than one parent node.

In this database model data is more related as more relationships are established in this database model. Also, as the data is more related, hence accessing the data is also easier and fast. This database model was used to map many-to-many data relationships.

This was the most widely used database model, before Relational Model was introduced.



Entity-relationship Model

In this database model, relationships are created by dividing object of interest into entity and its characteristics into attributes.

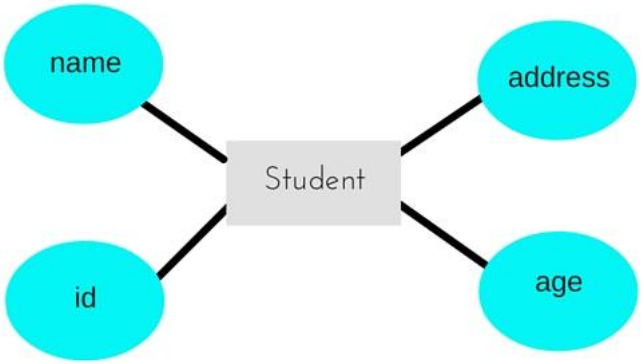
Different entities are related using relationships.

E-R Models are defined to represent the relationships into pictorial form to make it easier for different stakeholders to understand.

This model is good to design a database, which can then be turned into tables in relational model(explained below).

Let's take an example, If we have to design a School Database, then **Student** will be an **entity** with **attributes** name, age, address etc. As **Address** is generally complex, it can be another **entity** with **attributes** street name, pincode, city etc, and there will be a relationship between them.

Relationships can also be of different types. To learn about E-R Diagrams in details, click on the link.



Relational Model

In this model, data is organised in two-dimensional **tables** and the relationship is maintained by storing a common field.

This model was introduced by E.F Codd in 1970, and since then it has been the most widely used database model, infact, we can say the only database model used around the world.

The basic structure of data in the relational model is tables. All the information related to a particular type is stored in rows of that table.

Hence, tables are also known as **relations** in relational model.

In the coming tutorials we will learn how to design tables, normalize them to reduce data redundancy and how to use Structured Query language to access data from tables.

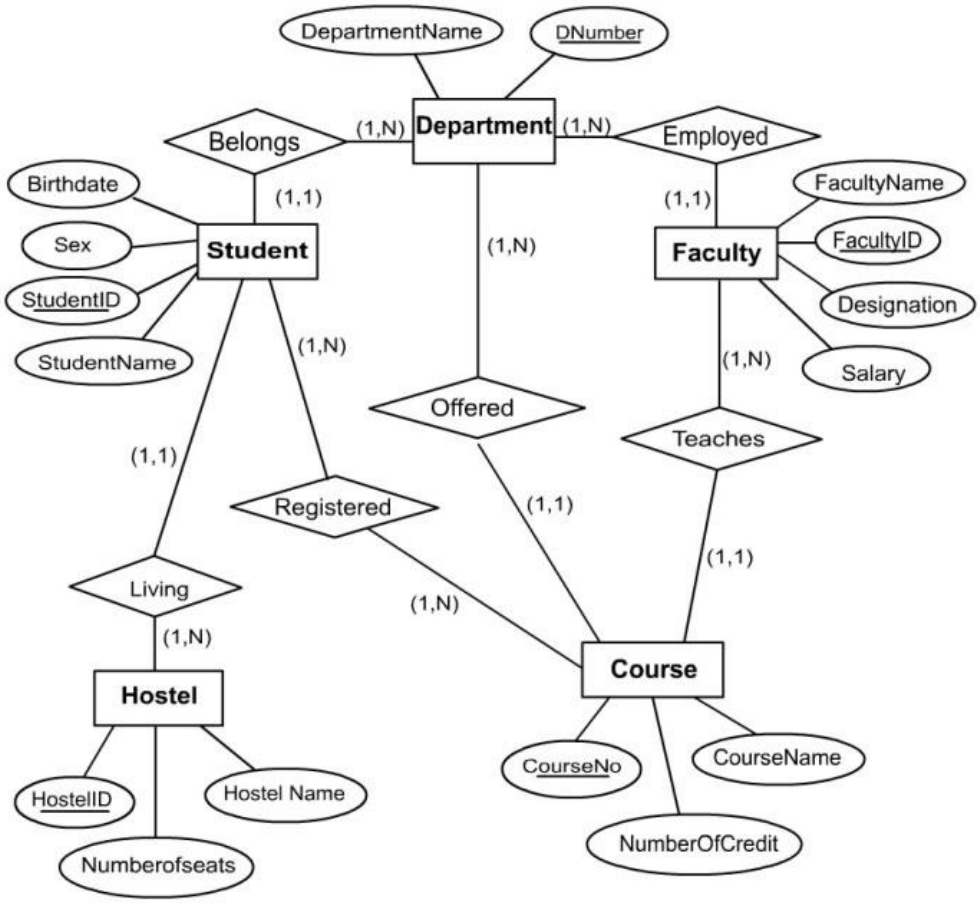
student_id	name	age
1	Akon	17
2	Bkon	18
3	Ckon	17
4	Dkon	18

subject_id	name	teacher
1	Java	Mr. J
2	C++	Miss C
3	C#	Mr. C Hash
4	Php	Mr. P H P

student_id	subject_id	marks
1	1	98
1	2	78
2	1	76
3	2	88

Q2 a) Draw and explain ER diagram for college Management system.

Ans-



Q2 b) What is significance of view? Also mention its syntax in SQL.

Ans-

A view is nothing more than a SQL statement that is stored in the database with an associated name. A view is actually a composition of a table in the form of a predefined SQL query.

A view can contain all rows of a table or select rows from a table. A view can be created from one or many tables which depends on the written SQL query to create a view.

Creating Views

Database views are created using the **CREATE VIEW** statement. Views can be created from a single table, multiple tables or another view.

To create a view, a user must have the appropriate system privilege according to the specific implementation.

The basic **CREATE VIEW** syntax is as follows –

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

Views, which are a type of virtual tables allow users to do the following –

- Structure data in a way that users or classes of users find natural or intuitive.
- Restrict access to the data in such a way that a user can see and (sometimes) modify exactly what they need and no more.
- Summarize data from various tables which can be used to generate reports.

Q2 c) What are the drawbacks of file processing system?

Ans-

The file processing system has the following major disadvantages:

- Data redundancy and inconsistency.
- Integrity Problems.
- Security Problems
- Difficulty in accessing data.
- Data isolation.

a) Data redundancy and inconsistency:

Data redundancy means duplication of data and inconsistency means that the duplicated values are different.

b) Integrity problems:

Data integrity means that the data values in the data base should be accurate in the sense that the value must satisfy some rules.

c) Security Problem:

Data security means prevention of data accession by unauthorized users.

d) Difficulty in accessing data:

Difficulty in accessing data arises whenever there is no application program for a specific task.

e) Data isolation:

This problem arises due to the scattering of data in various files with various formats. Due to the above disadvantages of the earlier data processing system, the necessity for an effective data processing system arises. Only at that time the concept of DBMS emerges for the rescue of a large number of organizations.

Q3 a) Discuss primary key and foreign key with suitable example.

Ans-

Primary Keys

In order for a table to qualify as a relational table it must have a primary key.

The **primary key** consists of one or more columns whose data contained within is used to uniquely identify each row in the table. You can think of the primary key as an address. If the rows in a table were mailboxes, then the primary key would be the listing of street addresses.

When a primary key is composed of multiple columns, the data from each column is used to determine whether a row is unique.

Ex

```
CREATE TABLE Persons (
    ID int NOT NULL PRIMARY KEY,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int
);
```

Foreign Keys

A **foreign key** is a set of one or more columns in a table that refers to the primary key in another table. There isn't any special code, configurations, or table definitions you need to place to officially "designate" a foreign key.

Ex

```
CREATE TABLE Orders (
    OrderID int NOT NULL PRIMARY KEY,
    OrderNumber int NOT NULL,
    PersonID int FOREIGN KEY REFERENCES Persons(PersonID)
);
```

Q3 b) Describe with suitable example in relational algebra

i) Union ii) Natural Join iii) Intersection iv) Set difference

Ans-

1) The UNION operator is used to combine the result-set of two or more SELECT statements.

   - Each SELECT statement within UNION must have the same number of columns
   - The columns must also have similar data types
   - The columns in each SELECT statement must also be in the same order

   UNION Syntax

   SELECT *column_name(s)* FROM *table1*
   UNION
   SELECT *column_name(s)* FROM *table2*;

2) A **NATURAL JOIN** is a **JOIN** operation that creates an implicit **join** clause for you based on the common columns in the two tables being **joined**. Common columns are columns that have the same name in both tables.

```
SELECT * FROM product NATURAL JOIN category;
```

3) The SQL INTERSECT operator is used to return the **results** of 2 or more SELECT statements. However, it only returns the rows selected by all **queries** or data sets. If a record exists in one query and not in the other, it will be omitted from the INTERSECT **results**.

The basic syntax of **INTERSECT** is as follows.

```
SELECT column1 [, column2 ] FROM table1 [, table2 ] [WHERE condition]

INTERSECT

SELECT column1 [, column2 ] FROM table1 [, table2 ] [WHERE condition]
```

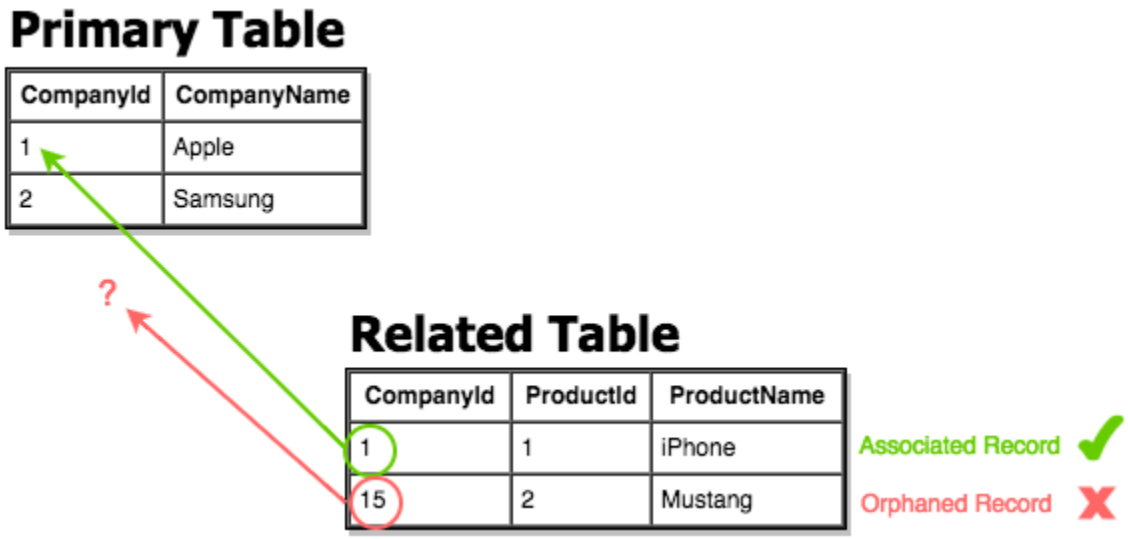
4) Set Difference in relational algebra is same set difference operation as in set theory with the constraint that both relation should have same set of attributes.

Q 4 a) What do you mean by referential integrity? How it is achieved in SQL?

Ans-

Referential integrity refers to the accuracy and consistency of data within a relationship. In relationships, data is linked between two or more tables. This is **achieved** by having the **foreign key** (in the associated table) reference a primary key value.

For example, if we delete record number 15 in a primary table, we need to be sure that there's no foreign key in any related table with the value of 15. We should only be able to delete a primary key if there are no associated records. Otherwise, we would end up with an orphaned record.



So referential integrity will prevent users from:

- Adding records to a related table if there is no associated record in the primary table.
- Changing values in a primary table that result in orphaned records in a related table.
- Deleting records from a primary table if there are matching related records.

Referential integrity is a subset of data integrity, which is concerned with the accuracy and consistency of all data (relationship or otherwise). Maintaining data integrity is a crucial part of working with databases.

Q4 b) Discuss any three aggregate function and any three string function with example.

Ans- Aggregate Functions are all about

- Performing calculations on multiple rows
- Of a single column of a table
- And returning a single value.

The aggregate functions namely;

1) COUNT
2) SUM
3) AVG

COUNT Function

The COUNT function returns the total number of values in the specified field. It works on both numeric and non-numeric data types.

```
SELECT COUNT(`movie_id`)  FROM `movierentals` WHERE `movie_id` = 2;
```

SUM function

Suppose we want a report that gives total amount of payments made so far. We can use the MySQL **SUM** function which **returns the sum of all the values in the specified column**. **SUM works on numeric fields only**. **Null values are excluded from the result returned.**

```
SELECT SUM(`amount_paid`) FROM `payments`;
```

AVG function

MySQL AVG function **returns the average of the values in a specified column**. Just like the SUM function, it **works only on numeric data types**.

```
SELECT AVG(`amount_paid`)  FROM `payments`;
```

String Function

LEN function

LEN function to determine a source string's length. It takes a single parameter containing a string expression.

```
SELECT LEN('This is string') AS Length
```

Length

14

LEFT

You use the LEFT function to return a specified number of characters from a string's left side. It takes two parameters: the source string expression and an integer indicating the number of characters to return.

```
SELECT LEFT ('SQL Server 2008', 3) As SQL
```

SQL

SQL

RIGHT

The RIGHT function returns a specified number of characters from a string's right side. It also accepts a string expression and an integer.

```
SELECT RIGHT ('SQL Server 2008',4) As Release
```

Release

2008

Q5 a) Define Normalization. Explain 1NF, 2NF, 3NF and BCNF with suitable example

Ans-

Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anamolies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables.

Normalization is used for mainly two purposes,

- Eliminating redundant (useless) data.
- Ensuring data dependencies make sense i.e data is logically stored.

First Normal Form (1NF)

For a table to be in the First Normal Form, it should follow the following 4 rules:

- It should only have single (atomic) valued attributes/columns.
- Values stored in a column should be of the same domain
- All the columns in a table should have unique names.
- And the order in which data is stored, does not matter.

Student	Age	Subject
Rooney	15	Java, C++
Kane	16	HTML, PHP

Normalized Table: Any Row must not have a column in which more than one value is saved, instead data is separated in multiple rows as shown below.

Student	Age	Subject
Rooney	15	JAVA
Rooney	15	C++
Kane	16	HTML
Kane	16	PHP

Second Normal Form (2NF)

For a table to be in the Second Normal Form,

1. It should be in the First Normal form.
2. And, it should not have Partial Dependency.

Stud_ID	Proj_ID	Stud_Name	Proj_Name
100	001	Rooney	Cloud
200	002	Kane	Servers

Stud_Name depends on Stud_ID and Proj_Name depends on Proj_ID

The above table can be normalized to 2NF as shown below.

Student Table in 2NF

Stud_ID	Proj_ID	Stud_Name
100	001	Rooney
200	001	Kane

Project Table in 2NF

Proj_ID	Proj_Name
001	001
002	Servers

Third Normal Form (3NF)

A table is said to be in the Third Normal Form when,

1. It is in the Second Normal form.
2. And, it doesn't have Transitive Dependency.

Stud_ID	Stud_Name	City	Zip
100	Rooney	Manchester	4001
200	Kane	Stoke	4002

Stud_ID is the only prime key attribute. City can be identified by Stu_ID as well as Zip. Neither Zip is a superkey nor City is a prime attribute.

Stud_ID -> Zip -> City, so there exists transitive dependency. Hence 3NF table is below

Student_Detail

Stud_ID	Stud_Name	Zip
100	Rooney	4001
200	Kane	4002

Zip_Code

Zip	City
4001	Manchester
4002	Stoke

Q 6a) Write short note on:

i) Primary and Secondary indexing.

Ans-

Indexing is a data structure technique to efficiently retrieve records from the database files based on some attributes on which the indexing has been done. Indexing in database systems is similar to what we see in books.

Indexing is defined based on its indexing attributes. Indexing can be of the following types –

- **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.
- **Secondary Index** – Secondary index may be generated from a field which is a candidate key and has a unique value in every record, or a non-key with duplicate values.

ii) Sparse and Dense indexing

Ans-

Ordered Indexing is of two types –

- Dense Index
- Sparse Index

Dense Index

In dense index, there is an index record for every search key value in the database. This makes searching faster but requires more space to store index records itself. Index records contain search key value and a pointer to the actual record on the disk.

Sparse Index

In sparse index, index records are not created for every search key. An index record here contains a search key and an actual pointer to the data on the disk. To search a record, we first proceed by index record and reach at the actual location of the data. If the data we are looking for is not where we directly reach by following the index, then the system starts sequential search until the desired data is found.

Q7 a) What is pipelining? Discuss its types.

Ans-

Pipelining is primarily used to create and organize a **pipeline** of instructions for a computer processor to processes in parallel. Typically, **pipelining** is an ongoing process where new tasks are added frequently and completed tasks are removed.

- With pipelined evaluation, operations form a queue, and results are passed from one operation to another as they are calculated, hence the technique's name.
- With pipelined evaluation, operations form a queue, and results are passed from one operation to another as
  they are calculated, hence the technique's name.
- General approach: restructure the individual operation algorithms so that they take streams of tuples as both input and output.
- So for instance, algorithms that require sorting can only use pipelining if the input is already sorted beforehand, since sorting by nature cannot be performed until all tuples to be sorted are known.

It is divided into 2 categories:

1. Arithmetic Pipeline
2. Instruction Pipeline

Arithmetic Pipeline

Arithmetic pipelines are usually found in most of the computers. They are used for floating point operations, multiplication of fixed point numbers etc. For example: The input to the Floating Point Adder pipeline is:

*X = A*2^a*

*Y = B*2^b*

Here A and B are mantissas (significant digit of floating point numbers), while **a** and **b** are exponents.

The floating point addition and subtraction is done in 4 parts:

1. Compare the exponents.
2. Align the mantissas.
3. Add or subtract mantissas
4. Produce the result.

Registers are used for storing the intermediate results between the above operations.

Instruction Pipeline

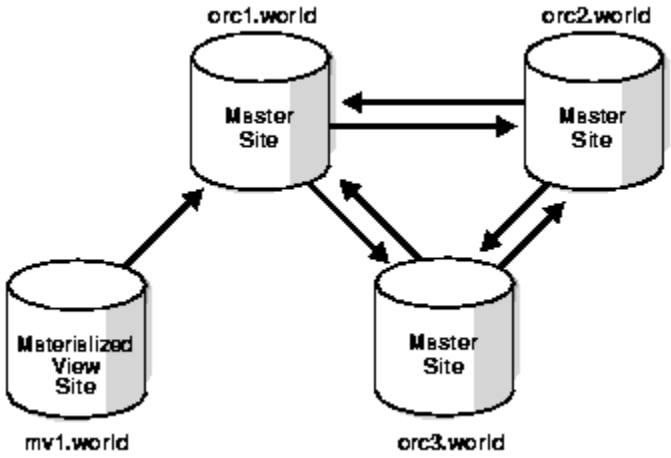
In this a stream of instructions can be executed by overlapping *fetch*, *decode* and *execute* phases of an instruction cycle. This type of technique is used to increase the throughput of the computer system.

An instruction pipeline reads instruction from the memory while previous instructions are being executed in other segments of the pipeline. Thus we can execute multiple instructions simultaneously. The pipeline will be more efficient if the instruction cycle is divided into segments of equal duration.

Q7 b) What is mean by Materialization? Explain it with the help of example

Ans-

Materialized view is a database object that contains the results of a query. ... Whenever a query or an update addresses an ordinary view's virtual table, the **DBMS** converts these into queries or updates against the underlying base tables.



Q8 a) What is Query optimization & its various techniques.

Ans-

- Query optimization is a difficult part of the query processing.
- It determines the efficient way to execute a query with different possible query plans.
- It cannot be accessed directly by users once the queries are submitted to the database server or parsed by the parser.
- A query is passed to the query optimizer where optimization occurs.
- Main aim of Query Optimization is to minimize the cost function, I/O Cost + CPU Cost + Communication Cost
- It defines how an RDBMS can improve the performance of the query by re-ordering the operations.
- It is the process of selecting the most efficient query evaluation plan from among various strategies if the query is complex.
- It computes the same result as per the given expression, but it is a least costly way of generating result.

Importance of Query Optimization

- Query optimization provides faster query processing.
- It requires less cost per query.
- It gives less stress to the database.
- It provides high performance of the system.
- It consumes less memory.

Q b) What is Query processing? Explain steps involved in query processing.

Ans-

- Query Processing is a translation of high-level queries into low-level expression.
- It is a step wise process that can be used at the physical level of the file system, query optimization and actual execution of the query to get the result.
- It requires the basic concepts of relational algebra and file structure.
- It refers to the range of activities that are involved in extracting data from the database.

- It includes translation of queries in high-level database languages into expressions that can be implemented at the physical level of the file system.
- In query processing, we will actually understand how these queries are processed and how they are optimized.

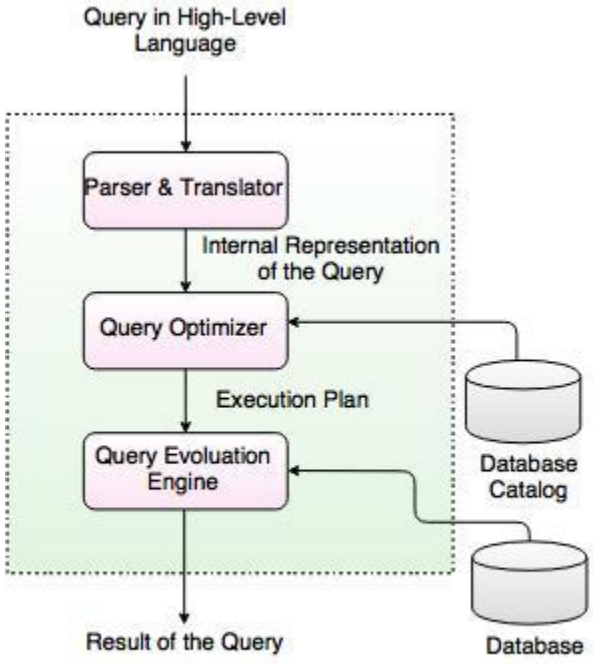


Fig. Query Processing

In the above diagram,

- The first step is to transform the query into a standard form.
- A query is translated into SQL and into a relational algebraic expression. During this process, Parser checks the syntax and verifies the relations and the attributes which are used in the query.
- The second step is Query Optimizer. In this, it transforms the query into equivalent expressions that are more efficient to execute.
- The third step is Query evaluation. It executes the above query execution plan and returns the result.

Example

```
SELECT Ename FROM Employee
  WHERE Salary > 5000;
```

Translated into Relational Algebra Expression

σ Salary > 5000 (π Ename (Employee))

OR

π Ename (σ Salary > 5000 (Employee))

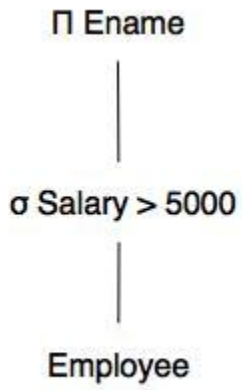


Fig. Query Execution Plan

- A sequence of primitive operations that can be used to evaluate a query is a Query Execution Plan or Query Evaluation Plan.
- The above diagram indicates that the query execution engine takes a query execution plan and returns the answers to the query.
- Query Execution Plan minimizes the cost of query evaluation.

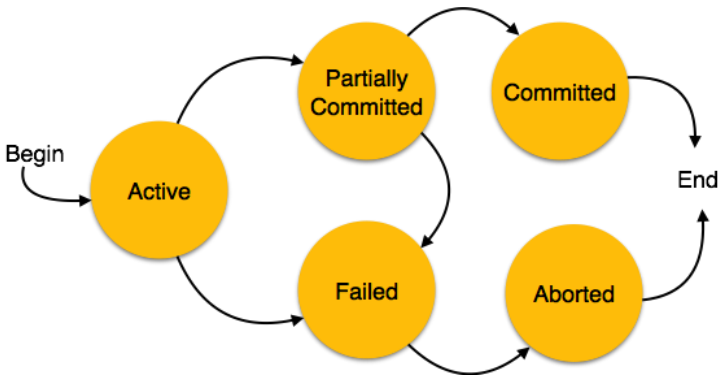
Q9 a) Define Transaction. What are the different states of transactions? Give ACID properties of transactions.

Ans-

A transaction is a very small unit of a program and it may contain several lowlevel tasks. A transaction in a database system must maintain **A**tomicity, **C**onsistency, **I**solation, and **D**urability – commonly known as ACID properties – in order to ensure accuracy, completeness, and data integrity.

States of Transactions

A transaction in a database can be in one of the following states –



- **Active** – In this state, the transaction is being executed. This is the initial state of every transaction.

- **Partially Committed** – When a transaction executes its final operation, it is said to be in a partially committed state.
- **Failed** – A transaction is said to be in a failed state if any of the checks made by the database recovery system fails. A failed transaction can no longer proceed further.
- **Aborted** – If any of the checks fails and the transaction has reached a failed state, then the recovery manager rolls back all its write operations on the database to bring the database back to its original state where it was prior to the execution of the transaction. Transactions in this state are called aborted. The database recovery module can select one of the two operations after a transaction aborts –
    - Re-start the transaction
    - Kill the transaction
- **Committed** – If a transaction executes all its operations successfully, it is said to be committed. All its effects are now permanently established on the database system.

Properties

- **Atomicity** – This property states that a transaction must be treated as an atomic unit, that is, either all of its operations are executed or none. There must be no state in a database where a transaction is left partially completed. States should be defined either before the execution of the transaction or after the execution/abortion/failure of the transaction.
- **Consistency** – The database must remain in a consistent state after any transaction. No transaction should have any adverse effect on the data residing in the database. If the database was in a consistent state before the execution of a transaction, it must remain consistent after the execution of the transaction as well.
- **Durability** – The database should be durable enough to hold all its latest updates even if the system fails or restarts. If a transaction updates a chunk of data in a database and commits, then the database will hold the modified data. If a transaction commits but the system fails before the data could be written on to the disk, then that data will be updated once the system springs back into action.
- **Isolation** – In a database system where more than one transaction are being executed simultaneously and in parallel, the property of isolation states that all the transactions will be carried out and executed as if it is the only transaction in the system. No transaction will affect the existence of any other transaction.

Q9 b) Explain two phase commit protocol in detail.

Ans-

Two-phase commit reduces the vulnerability of one-phase commit protocols. The steps performed in the two phases are as follows –

Phase 1: Prepare Phase

- After each slave has locally completed its transaction, it sends a "DONE" message to the controlling site. When the controlling site has received "DONE" message from all slaves, it sends a "Prepare" message to the slaves.
- The slaves vote on whether they still want to commit or not. If a slave wants to commit, it sends a "Ready" message.
- A slave that does not want to commit sends a "Not Ready" message. This may happen when the slave has conflicting concurrent transactions or there is a timeout.

Phase 2: Commit/Abort Phase

- After the controlling site has received "Ready" message from all the slaves –
    - The controlling site sends a "Global Commit" message to the slaves.
    - The slaves apply the transaction and send a "Commit ACK" message to the controlling site.
    - When the controlling site receives "Commit ACK" message from all the slaves, it considers the transaction as committed.
- After the controlling site has received the first "Not Ready" message from any slave –
    - The controlling site sends a "Global Abort" message to the slaves.
    - The slaves abort the transaction and send a "Abort ACK" message to the controlling site.
    - When the controlling site receives "Abort ACK" message from all the slaves, it considers the transaction as aborted.

Q10 a) What is serializability? Explain conflict & view serializability.

Ans-

When multiple transactions are being executed by the operating system in a multiprogramming environment, there are possibilities that instructions of one transactions are interleaved with some other transaction.

- **Schedule** – A chronological execution sequence of a transaction is called a schedule. A schedule can have many transactions in it, each comprising of a number of instructions/tasks.
- **Serial Schedule** – It is a schedule in which transactions are aligned in such a way that one transaction is executed first. When the first transaction completes its cycle, then the next transaction is executed. Transactions are ordered one after the other. This type of schedule is called a serial schedule, as transactions are executed in a serial manner.

In a multi-transaction environment, serial schedules are considered as a benchmark. The execution sequence of an instruction in a transaction cannot be changed, but two transactions can have their instructions executed in a random fashion. This execution does no harm if two transactions are mutually independent and working on different segments of data; but in case these two transactions are working on the same data, then the results may vary. This ever-varying result may bring the database to an inconsistent state.

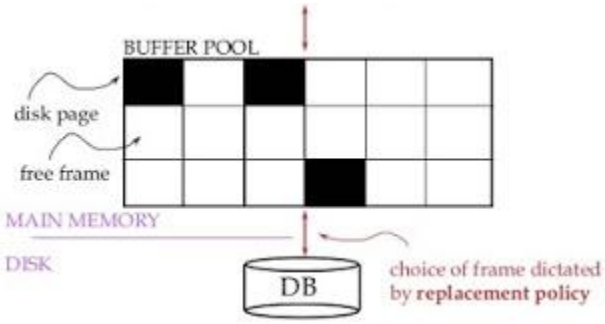
Q10 b) What are the different buffer management techniques?

Ans-

BUFFER MANAGEMENT

The **buffer manager** is the software layer that is responsible for bringing pages from physical disk to main memory as needed. The buffer manages the available main memory by dividing the main memory into a collection of pages, which we called as **buffer pool.** The main memory pages in the buffer pool are called **frames.**

Page Requests from Higher Level



The goal of the buffer manager is to ensure that the data requests made by programs are satisfied by copying data from secondary storage devices into buffer. Infact, if a program performs reading from existing buffers. similarly, if a program performs an output statement: it calls the buffer manager for output operation - to satisfy the requests by writing to the buffers. Therefore, we can say that input and output operation occurs between the program and the buffer area only.

if an input operation does not find the requested page in the buffer pool, then the buffer manager (software) will have to do a physical transfer the page from the secondary memory (disk) to a free block in buffer pool and then make the requested page placed in the buffer pool, that is, available to the program requesting the original input operation. A similar scenario will take place in the reverse order for an output operation. That is, the buffer manager (software) makes a new empty buffer a available to the program for outputting the page. If there is no space in the buffer pool then the buffer manager (software) physically transfer one of the page from buffer pool to the disk (secondary memory) to provide the empty space in the buffer pool for the output operation of the program to display the page in the buffer pool.

Q11 a) Describe different types of failures that occurs in the system? How they are recovered.

Ans-

A distributed database system, failures can be broadly categorized into soft failures, hard failures and network failures.

Soft Failure

Soft failure is the type of failure that causes the loss in volatile memory of the computer and not in the persistent storage. Here, the information stored in the non-persistent storage like main memory, buffers, caches or registers, is lost. They are also known as system crash. The various types of soft failures are as follows –

- Operating system failure.
- Main memory crash.

- Transaction failure or abortion.
- System generated error like integer overflow or divide-by-zero error.
- Failure of supporting software.
- Power failure.

Hard Failure

A hard failure is the type of failure that causes loss of data in the persistent or non-volatile storage like disk. Disk failure may cause corruption of data in some disk blocks or failure of the total disk. The causes of a hard failure are –

- Power failure.
- Faults in media.
- Read-write malfunction.
- Corruption of information on the disk.
- Read/write head crash of disk.

Recovery from disk failures can be short, if there is a new, formatted, and ready-to-use disk on reserve. Otherwise, duration includes the time it takes to get a purchase order, buy the disk, and prepare it.

Network Failure

Network failures are prevalent in distributed or network databases. These comprises of the errors induced in the database system due to the distributed nature of the data and transferring data over the network. The causes of network failure are as follows –

- Communication link failure.
- Network congestion.
- Information corruption during transfer.
- Site failures.
- Network partitioning.

Q11 b) Write a short note on checkpoint.

Ans-

Checkpoint

• Keeping and maintaining logs in real time and in real environment may fill out all the memory space available in the system.

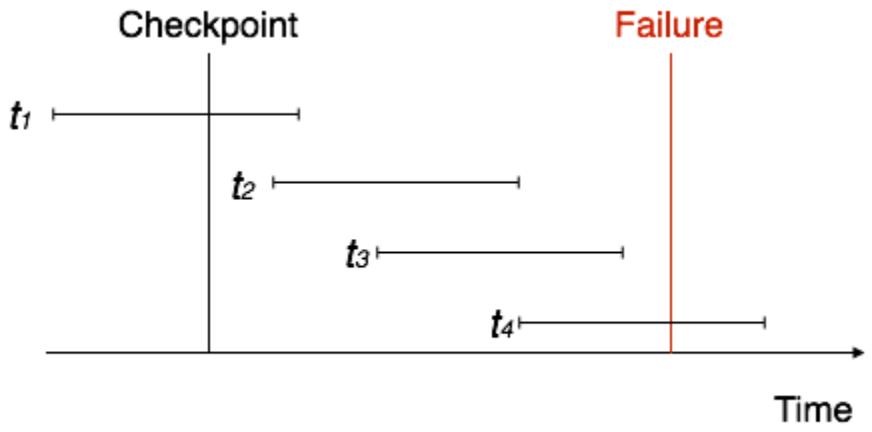
• The log file may grow too big to be handled as time passes.

• 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.

Recovery

When a system with concurrent transactions crashes and recovers, it behaves in the following manner –



• The recovery system reads the logs backwards from the end to the last checkpoint.

• It maintains two lists, an undo-list and a redo-list.

• If the recovery system sees a log with <tn, start=""> and <tn, commit=""> or just <tn, commit="">, it puts the transaction in the redo-list.

• If the recovery system sees a log with <tn, start=""> but no commit or abort log found, it puts the transaction in undo-list.

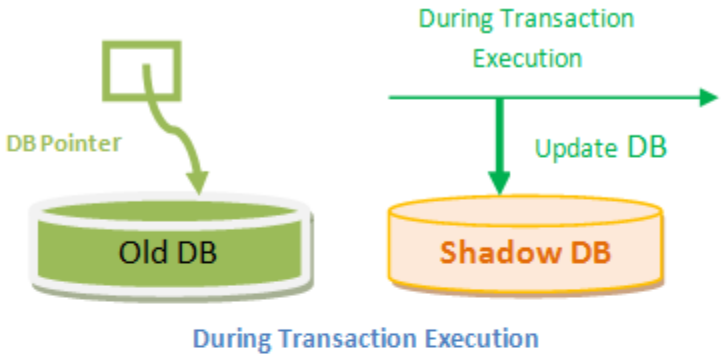
• All the transactions in the undo-list are then undone and their logs are removed.

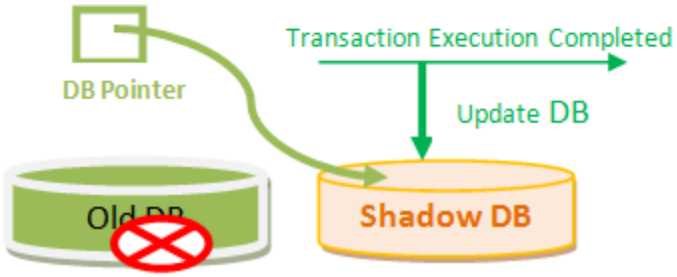
• All the transactions in the redo-list and their previous logs are removed and then redone before saving their logs.

Q11 c) Write a short note on shadow paging

Ans-

This is the method where all the transactions are executed in the primary memory or the shadow copy of database. Once all the transactions completely executed, it will be updated to the database. Hence, if there is any failure in the middle of transaction, it will not be reflected in the database. Database will be updated after all the transaction is complete.





A database pointer will be always pointing to the consistent copy of the database, and copy of the database is used by transactions to update. Once all the transactions are complete, the DB pointer is modified to point to new copy of DB, and old copy is deleted. If there is any failure during the transaction, the pointer will be still pointing to old copy of database, and shadow database will be deleted. If the transactions are complete then the pointer is changed to point to shadow DB, and old DB is deleted.

Q12 Write a short note on **any three.**

i) Distributed database.

Ans-

A distributed database is a collection of multiple interconnected databases, which are spread physically across various locations that communicate via a computer network.

Features

- Databases in the collection are logically interrelated with each other. Often they represent a single logical database.
- Data is physically stored across multiple sites. Data in each site can be managed by a DBMS independent of the other sites.
- The processors in the sites are connected via a network. They do not have any multiprocessor configuration.
- A distributed database is not a loosely connected file system.
- A distributed database incorporates transaction processing, but it is not synonymous with a transaction processing system.

Advantages of Distributed Databases

Following are the advantages of distributed databases over centralized databases.

Modular Development − If the system needs to be expanded to new locations or new units, in centralized database systems, the action requires substantial efforts and disruption in the existing functioning. However, in distributed databases, the work simply requires adding new computers and local data to the new site and finally connecting them to the distributed system, with no interruption in current functions.

More Reliable − In case of database failures, the total system of centralized databases comes to a halt. However, in distributed systems, when a component fails, the functioning of the system continues may be at a reduced performance. Hence DDBMS is more reliable.

Better Response − If data is distributed in an efficient manner, then user requests can be met from local data itself, thus providing faster response. On the other hand, in centralized systems, all queries have to pass through the central computer for processing, which increases the response time.

Lower Communication Cost − In distributed database systems, if data is located locally where it is mostly used, then the communication costs for data manipulation can be minimized. This is not feasible in centralized systems.

ii) Web database.

Ans-

A Web database is a database application designed to be managed and accessed through the Internet. Website operators can manage this collection of data and present analytical results based on the data in the Web database application. Databases first appeared in the 1990s, and have been an asset for businesses,

allowing the collection of seemingly infinite amounts of data from infinite amounts of customers.

Web database software programs are found within desktop publishing programs, such as Microsoft Office Access and OpenOffice Base. Other programs include the Webex WebOffice database and FormLogix Web database. The most advanced software applications can set up data collection forms, polls, feedback forms and present data analysis in real time.

iii) Data warehouse

Ans-

Data warehousing is the process of constructing and using a data warehouse. A data warehouse is constructed by integrating data from multiple heterogeneous sources that support analytical reporting, structured and/or ad hoc queries, and decision making. Data warehousing involves data cleaning, data integration, and data consolidations.

The following are the functions of data warehouse tools and utilities –

- Data Extraction – Involves gathering data from multiple heterogeneous sources.
- Data Cleaning – Involves finding and correcting the errors in data.
- Data Transformation – Involves converting the data from legacy format to warehouse format.
- Data Loading – Involves sorting, summarizing, consolidating, checking integrity, and building indices and partitions.
- Refreshing – Involves updating from data sources to warehouse.

iv) Data Mining.

Ans- Data Mining is defined as extracting information from huge sets of data. In other words, we can say that data mining is the procedure of mining knowledge from data. The information or knowledge extracted so can be used for any of the following applications –

- Market Analysis
- Fraud Detection
- Customer Retention
- Production Control

- Science Exploration

Data Mining Applications

Data mining is highly useful in the following domains –

- Market Analysis and Management
- Corporate Analysis & Risk Management
- Fraud Detection

Apart from these, data mining can also be used in the areas of production control, customer retention, science exploration, sports, astrology, and Internet Web Surf-Aid

Market Analysis and Management

Listed below are the various fields of market where data mining is used –

- **Customer Profiling** – Data mining helps determine what kind of people buy what kind of products.
- **Identifying Customer Requirements** – Data mining helps in identifying the best products for different customers. It uses prediction to find the factors that may attract new customers.
- **Cross Market Analysis** – Data mining performs Association/correlations between product sales.
- **Target Marketing** – Data mining helps to find clusters of model customers who share the same characteristics such as interests, spending habits, income, etc.
- **Determining Customer purchasing pattern** – Data mining helps in determining customer purchasing pattern.
- **Providing Summary Information** – Data mining provides us various multidimensional summary reports.