Module 1

Purpose of Database System:

A database system is like a digital filing cabinet for storing and managing data. It's used to store, organize, and retrieve information efficiently. The primary purposes of a database system are to provide a structured way to store data, ensure data integrity, allow for easy data retrieval and manipulation, and support multiple users accessing and sharing data.

Views of Data:

Different users may have different perspectives or views of the data in a database. For example, a salesperson might only be interested in customer information, while an accountant might focus on financial data. Views of data allow users to see and work with the parts of the database that are relevant to their tasks without having to access the entire database.

Data Models:

Data models are like blueprints that define how data is organized and structured in a database. They provide a way to represent real-world entities and their relationships in a logical and organized manner. Common data models include the relational model

(tables), hierarchical model (tree-like structure), and object-oriented model (objects and classes).

Database System Architecture:

This refers to the overall design and structure of a database system. It includes components like the database management system (DBMS), the physical storage of data, and the user interfaces. The architecture ensures that data is stored efficiently and can be accessed securely.

Database Users and Administrator:

Users are people or programs that interact with the database, such as data entry clerks, analysts, and customers. Administrators are responsible for managing and maintaining the database, including tasks like setting up user accounts, ensuring data security, and optimizing performance.

Entity-Relationship Model (E-R Model):

The E-R model is a way to represent entities (things or objects) and their relationships in a database. Entities are represented as rectangles, and relationships are shown as lines connecting these rectangles. This model helps in designing the structure of a database.

E-R Diagrams:

E-R diagrams are visual representations of the E-R model. They use symbols like rectangles, diamonds, and lines to show entities, attributes, and relationships. E-R diagrams make it easier to understand the database's structure.

Cardinality Relation:

Cardinality relation in the context of E-R diagrams defines how many instances of one entity can be related to instances of another entity. It includes concepts like one-to-one, one-to-many, and many-to-many relationships.

Types of Cardinality:

One-to-One: Each instance in one entity is related to exactly one instance in another entity.

One-to-Many: Each instance in one entity can be related to multiple instances in another entity.

Many-to-Many: Multiple instances in one entity can be related to multiple instances in another entity.

Introduction to Relational Databases:

Relational databases use tables to store data, where each table represents an entity, and rows represent individual records. They are based on the

principles of the relational model, which was developed by Dr. EF Codd.

Dr. EF Codd Rules Relational Model:

Dr. EF Codd established a set of rules (Codd's rules) that a database must follow to be considered a true relational database. These rules ensure data integrity, consistency, and reliability.

Database Languages - SQL Fundamentals:

SQL (Structured Query Language) is a language used to interact with relational databases. It includes several sub-languages for different purposes.

DRL (Data Retrieval Language): Used for querying data from the database.

DML (Data Manipulation Language): Used for adding, modifying, and deleting data.

TCL (Transaction Control Language): Used to manage transactions.

DDL (Data Definition Language): Used to define the structure of the database.

DCL (Data Control Language): Used for managing access control and permissions.

DRL Language - SELECT Syntax, Literal, Command, Logical Operator, Range Searching, Pattern Matching,

Order By Clause, Group By Clause:

SELECT Syntax: It is the basic SQL command for retrieving data from a database.

Literal: A literal is a fixed value like a number or text string.

Command: SQL commands are instructions to perform specific tasks.

Logical Operator: Operators like AND, OR, and NOT are used to combine conditions.

Range Searching: You can search for data within a specified range using operators like BETWEEN.

Pattern Matching: Used to search for data based on patterns using wildcards like % or _.

Order By Clause: Sorts query results in a specified order.

Group By Clause: Groups query results based on a specified column, often used with aggregate functions like SUM or COUNT.

In summary, a database system is a structured way to store and manage data, with various components and user roles. The E-R model helps in designing the database structure, and SQL is used to interact with relational databases using different sub-languages for various tasks.

Module 2

Normalization:

Normalization is a process in database design that helps organize data efficiently and reduce redundancy. It involves breaking down large tables into smaller, related tables and establishing relationships between them. This ensures that each piece of data is stored in only one place, making the database more efficient and preventing data inconsistencies. There are different levels of normalization, including First, Second, Third, and even Fourth Normal Forms, each with specific rules to ensure data integrity and minimize duplication.

First Normal Form (1NF): Ensures that each column in a table contains only atomic (indivisible) values, and there are no repeating groups or arrays.

Second Normal Form (2NF): Builds on 1NF and ensures that each non-key column is fully functionally dependent on the entire primary key.

Third Normal Form (3NF): Builds on 2NF and ensures that there are no transitive dependencies, meaning that non-key attributes are not dependent on other non-key attributes.

Boyce/Codd Normal Form (BCNF): A more stringent form of normalization that ensures that for every non-

trivial functional dependency, the left-hand side is a superkey.

Fourth Normal Form (4NF): Addresses multi-valued dependencies, where one attribute's values are dependent on another attribute but not on the primary key.

Transactions:

In the context of databases, a transaction is a sequence of one or more database operations that are treated as a single unit of work. Transactions are used to ensure data consistency and integrity in a database system. Here are some important concepts related to transactions:

Transaction Concepts: Transactions are like a set of instructions that can include actions like inserting, updating, or deleting data. These actions are grouped together and treated as one unit.

Transaction Recovery: This refers to the ability of a database system to recover data to a consistent state in case of system failures or errors during a transaction. It ensures that even if something goes wrong, the database remains in a reliable state.

ACID Properties: ACID stands for Atomicity,

Consistency, Isolation, and Durability. These properties ensure that transactions are reliable and dependable. Atomicity means that a transaction is all or nothing, Consistency ensures that the database is always in a valid state, Isolation ensures that transactions don't interfere with each other, and Durability ensures that committed transactions are permanent.

Serializability: This is a concept that ensures that the execution of multiple transactions in a database is equivalent to some serial order of those transactions, even if they execute concurrently. It prevents data anomalies and maintains data consistency.

Types of Serializability: There are two common types: Conflict Serializability and View Serializability.

Conflict Serializability: It ensures that if there is a conflict (e.g., two transactions try to access the same data simultaneously), the database system will resolve it correctly.

View Serializability: It focuses on ensuring that the end result of concurrent transactions is the same as if they were executed one after the other.

In essence, transactions in databases ensure that data changes are made reliably and that the database

remains consistent, even in the face of unexpected events or concurrent access by multiple users or processes.

Module 3

Built-in Functions in SQL:

In SQL, built-in functions are like pre-defined tools that help you perform various operations on your data. These functions can be categorized into several types:

Single Row Functions: These functions work on individual rows of data and return a single result. For example, you can use functions to manipulate text, perform calculations, or change the format of dates.

Multiple Row Functions: These functions work on multiple rows of data and return a single result. They're often used with aggregate functions to summarize data across multiple rows.

Number Functions: Functions that perform operations on numeric data, such as addition, subtraction, and rounding.

Character Functions: Functions that work with text data, allowing you to manipulate strings, change case, or extract substrings.

Date Functions: Functions for working with date and time data, such as formatting, adding or subtracting time, and finding the current date.

Conversion Functions: These functions help you convert data from one data type to another.

General Functions: A catch-all category for functions that don't fit neatly into the other categories.

Grouping/Aggregate Functions:
These functions are used to perform calculations on groups of rows rather than individual rows. Common aggregate functions include SUM (adding values), AVG (calculating the average), COUNT (counting rows), MAX (finding the maximum value), and MIN (finding the minimum value). These are often used with the GROUP BY clause to group data and perform calculations within those groups.

Grouping Data from Tables in SQL:

The GROUP BY clause is used to group rows from a table based on a specified column. This allows you to perform aggregate functions on each group, creating summary data.

Joins:

Joins are used to combine data from two or more tables in a database. They are typically based on a related column between the tables. Joins help retrieve data from multiple tables to create a unified result set.

Joining Multiple Tables:

You can join more than two tables together to retrieve data from multiple sources in a single query. This is helpful when you have complex data relationships.

Different Types of Joins:

There are various types of joins, including INNER JOIN (returns only matching rows), LEFT JOIN (returns all rows from the left table and matching rows from the right table), RIGHT JOIN (returns all rows from the right table and matching rows from the left table), and FULL JOIN (returns all rows when there is a match in either table).

Subqueries:

A subquery is a query nested within another query. It's used to retrieve data that will be used as part of the main query. Subqueries can be used to filter, calculate, or retrieve specific data.

Types of Subqueries:

Single-Row Subquery: Returns a single value, often used in conditions or comparisons.

Multiple-Row Subquery: Returns multiple rows of data, which can be compared to a set of values.

Top N Analysis:

Top N analysis is a technique used to retrieve the top (or bottom) N rows of data based on specific criteria. It's commonly used to find, for example, the top-selling products or the highest-paid employees.

DML Statements (Insert, Update, Delete, Merge):

DML stands for Data Manipulation Language. These statements are used to modify data in a database.

INSERT: Adds new rows of data into a table.

UPDATE: Modifies existing data in a table.

DELETE: Removes data from a table.

MERGE: Combines INSERT, UPDATE, and DELETE

operations into a single statement, typically used for data synchronization.

Oracle Data Types:

Oracle, like other databases, has various data types that define the kind of data a column can hold, such as NUMBER for numeric data, VARCHAR2 for text, and DATE for date and time values.

TCL Commands (Commit, Rollback, Savepoint):

TCL stands for Transaction Control Language, and it deals with managing transactions.

COMMIT: Saves all changes made during a transaction to the database.

ROLLBACK: Undoes all changes made during a transaction, reverting the database to its previous state.

SAVEPOINT: Sets a point in the transaction to which you can later roll back if needed, allowing for partial rollbacks.

In summary, SQL offers a wide range of built-in functions for data manipulation, aggregation, and analysis. Joins help you combine data from multiple tables, and subqueries provide a way to retrieve specific data within a query. DML statements allow you to insert, update, delete, or merge data, while Oracle data types define the kind of data a column can hold. Finally,

TCL commands help you manage transactions by committing changes, rolling back to a previous state, or creating savepoints.

Module 4

Set Operators (UNION, INTERSECT, MINUS):

Set operators in SQL are used to combine the results of two or more SELECT statements. These operators treat the results as sets (collections of rows) and perform set operations on them:

UNION: Combines the result sets of two SELECT statements and removes duplicate rows.

INTERSECT: Returns only the rows that are common to both result sets of two SELECT statements.

MINUS (or EXCEPT in some database systems): Returns the rows that are present in the first result set but not in the second result set.

DDL Commands (CREATE, DROP, ALTER, RENAME, TRUNCATE):

DDL stands for Data Definition Language, and these commands are used to manage the structure and schema of a database:

CREATE: Used to create new database objects like tables, views, or indexes.

DROP: Deletes database objects such as tables, views, or indexes, along with all associated data.

ALTER: Modifies the structure of existing database objects, such as adding or removing columns from a table.

RENAME: Renames an existing database object, such as changing the name of a table.

TRUNCATE: Removes all rows from a table, but unlike DELETE, it does not log individual row deletions, making it faster for large tables.

Data Types:

Data types in a database define the kind of data that can be stored in a column. Common data types include INTEGER (for whole numbers), VARCHAR (for text), DATE (for dates), and BOOLEAN (for true/false values). Data types ensure data consistency and help the database system manage storage efficiently.

TABLE Command:

The TABLE command is used to create a new table in a database. You specify the table name and define the columns it will contain along with their data types.

Constraints:

Constraints are rules that you can apply to columns in a table to ensure data integrity and accuracy.

Common constraints include:

Primary Key: Ensures that each row has a unique identifier and is used to identify records.

Foreign Key: Establishes relationships between tables by referring to the primary key in another table.

Check: Specifies a condition that must be met for the data in a column.

Not Null: Ensures that a column cannot contain NULL values.

Unique: Ensures that all values in a column are unique within the table.

Setting Constraints Using Column-Level & Table-Level Constraints:

You can define constraints at either the column level or the table level:

Column-Level Constraints: Constraints are defined when you create a column, specifying rules for that specific column.

Table-Level Constraints: Constraints are defined separately from the column definitions, applying rules to one or more columns as a whole.

Defining Different Constraints on the Table:

A table can have multiple constraints defined on its columns. For example, you can have a primary key constraint to enforce uniqueness and a check constraint to ensure data meets specific criteria.

Defining Integrity Constraints in the ALTER:

Integrity constraints can also be added or modified after a table is created using the ALTER command. This allows you to refine the rules and constraints on your data as your database evolves.

In summary, set operators in SQL allow you to combine and manipulate result sets. DDL commands are used to manage the structure of the database, including creating, altering, or deleting objects. Data types define the kind of data that can be stored in columns. Constraints are rules applied to columns to ensure data integrity, and they can be defined at the column or table level. Constraints can also be added or modified using the ALTER command as needed.

Module 5

Views:

Views in a database are like virtual tables that don't store data themselves but provide a way to access data from one or more tables. They are used for simplifying complex queries and controlling access to data.

Create View: You can create a view by defining a SELECT statement that retrieves data from one or more tables. The view acts as a window through which you can query the data.

Types of View: There are two main types of views: simple views (based on a single table) and complex views (based on multiple tables or other views).

Operations on Views: Views support operations like SELECT, which allows you to query the data in the view. You can't directly insert, update, or delete data through a view unless it's an "updatable view" that meets specific criteria.

Dictionaries:

Dictionaries in the context of a database are often referred to as system catalogs or system tables. These

tables store metadata about the database schema, such as information about tables, views, users, and constraints.

Dictionaries for Table, View, User, Constraints, etc.: Each of these elements in a database has associated system tables or dictionaries that store information about them. For example, the table dictionary might store details about the table's columns and data types.

Creating a View:

To create a view, you write a SQL query that specifies the data you want to include in the view. The view is then stored in the database, and you can query it as if it were a regular table.

Renaming the Column of a View:

You can rename the columns of a view by giving them new names in the SELECT statement that defines the view. This doesn't change the original table's column names, only how they appear in the view.

Updating, Selection, Destroying a View:

Updating: In some cases, you can update data through a view if the view meets certain criteria, such as having a single base table and no complex expressions

in the SELECT statement.

Selection: You can query data from a view using a SELECT statement, just like you would with a regular table.

Destroying (Dropping): To remove a view, you use the DROP VIEW statement. This doesn't delete the underlying data; it just removes the view.

Creating Indexes:

Indexes in a database are like tables of contents for data tables. They improve query performance by allowing the database to quickly locate specific rows of data.

DCL (Data Control Language):

DCL commands control access to the database and are used for security and authorization.

Granting Permissions: The GRANT command is used to give specific privileges or permissions to users or roles. For example, you can grant SELECT permission on a table to a user.

Revoking Permissions: The REVOKE command is used to take away previously granted permissions.

Creating and Managing Users:

In a database, users are individual accounts that can access the database. You can create, modify, and delete user accounts as needed.

Create User: To create a user, you use the CREATE USER statement. This defines the user's login credentials and privileges.

Create Role: A role is like a group of users with similar permissions. You can create roles and assign users to them.

Provide Privileges: After creating users or roles, you grant them privileges using the GRANT command to specify what they can do in the database.

Triggers:

Triggers are special database objects that automatically execute in response to specific events, such as data changes in a table.

Syntax: A trigger has a specific syntax that includes the event that triggers it, the timing (e.g., before or after the event), and the action it performs.

Enabling/Disabling Triggers: You can enable or

disable triggers as needed. Disabling a trigger means it won't run until you enable it again.

In summary, views provide a way to access and manipulate data in a database, even though they don't store the data themselves. Dictionaries store metadata about database objects. Creating a view involves defining a SELECT statement. You can rename columns in a view, query views, and drop them when no longer needed. Indexes improve query performance. DCL commands manage user access and permissions. Triggers automatically respond to database events, and you can enable or disable them as required.