Module-4-oops

1What is SQL, and why is it essential in database management?

SQL stands for **Structured Query Language**.

It is a **standard language** used to interact with a **Relational Database Management System (RDBMS)** such as MySQL, Oracle, SQL Server, or PostgreSQL.

SQL helps you **store, retrieve, manage, and manipulate data** in a structured way using queries.

Why

Data Retrieval → SQL allows us to quickly search and fetch specific data (e.g., find all customers from Mumbai).

Data Manipulation → You can insert, update, delete records easily.

Data Definition → You can create, modify, or delete database structures like tables, views, and indexes.

Data Control → SQL provides permissions & security (who can access or change data).

Standardized Language → It's the universal language across almost all RDBMS, so learning SQL makes it easier to work with any database.

Efficient Data Management → Handles large volumes of data that would be impossible to manage manually**.**

2.Explain the difference between DBMS and RDBMS.

The following are the important differences between DBMS and RDBMS-

Key	**DBMS**	**RDBMS**

Definition	DBMS stands for Database Management System.	RDBMS stands for Relational Database Management System.
Data Storage	Data is stored as file.	Data is stored as tables.
Data Access	In DBMS, each data elements are to be accessed individually.	In RDBMS, multiple data elements can be accessed at same time.
Relationship	There is no relationship between data in DBMS.	Data is present in multiple tables which can be related to each other.
Normalization	Normalization cannot be achieved.	Normalization can be achieved.
Distributed database	DBMS has no support for distributed databases.	RDBMS supports distributed databases.
Data Quantity	DBMS deals with small quantity of data.	RDBMS deals with large quantity of data.
Data Redundancy	Data Redundancy is common in DBMS.	Data Redundancy can be reduced using key and indexes in RDBMS.
User	DBMS supports single user at a time.	RDBMS supports multiple users at a time.
Security	DBMS provides low security during data manipulation.	RDBMS has multilayer security during data manipulation.
Example	File systems, XML, etc.	Oracle, SQL Server.

3.Describe the role of SQL in managing relational databases.

What is SQL?

SQL or Structured Query Language is a standard programming language used for managing and manipulating relational databases. It allows users to interact with databases by performing various operations such as:

1.Querying Data: It Retrieving specific data from one or more tables using commands like SELECT.

2.Inserting Data: Adding new records to a database table with commands like INSERT.

3.Updating Data: Modifying existing records using commands like UPDATE.

4.Deleting Data: Removing records from a table with commands like DELETE.

5.Creating and Altering Tables: Defining new tables or modifying existing ones using commands like CREATE TABLE and ALTER TABLE.

6.Managing Database Structures: Handling database schemas, indexes, and constraints to ensure data integrity and optimize performanceCreating and Altering Tables: Defining new tables or modifying existing ones using commands like CREATE TABLE and ALTER TABLE.

4. What are the key features of SQL?

key features of SQL (Structured Query Language):

1. **Data Definition Language (DDL):**
   - SQL allows you to define and manage the structure of a database.

- Commands: CREATE, ALTER, DROP, TRUNCATE.

2. **Data Manipulation Language (DML):**
   - SQL lets you insert, update, delete, and retrieve data from tables.
   - Commands: INSERT, UPDATE, DELETE, SELECT.
3. **Data Control Language (DCL):**
   - SQL manages user access and permissions in the database.
   - Commands: GRANT, REVOKE.
4. **Transaction Control Language (TCL):**
   - SQL supports transaction management to ensure data integrity.
   - Commands: COMMIT, ROLLBACK, SAVEPOINT.
5. **Data Retrieval (Powerful Queries):**
   - The SELECT statement is highly flexible for retrieving data.
   - Supports filtering (WHERE), sorting (ORDER BY), grouping (GROUP BY), and conditions (HAVING).
6. **Joins and Relationships:**
   - SQL can combine data from multiple tables using joins (INNER, LEFT, RIGHT, FULL).
   - Supports primary keys and foreign keys to enforce relationships.
7. **Functions and Expressions:**
   - SQL provides built-in functions (e.g., COUNT, SUM, AVG, MIN, MAX) for calculations and aggregations.
   - Supports string, date, and mathematical functions.
8. **Portability:**

- SQL is a standard language and can be used with most relational database systems (MySQL, Oracle, SQL Server, PostgreSQL, etc.) with minimal changes.

9. **Security:**
   - Provides role-based access control, user authentication, and privileges for data protection.
10. **Data Integrity:**
    - Ensures accuracy and consistency of data using constraints (PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, CHECK, DEFAULT).
11. **Scalability and Flexibility:**
    - Can handle large volumes of data and complex queries efficiently.
    - Supports views, stored procedures, and triggers for reusability.

5. What are the basic components of SQL syntax?

basic components of SQL syntax (without examples):

1. **Clauses** – Keywords that define SQL operations.
2. **Statements** – Complete instructions made of clauses.
3. **Expressions** – Combinations of values, operators, and functions evaluated to produce a result.
4. **Predicates** – Conditions that return TRUE, FALSE, or UNKNOWN.
5. **Identifiers** – Names of database objects like tables, columns, or aliases.
6. **Keywords / Reserved Words** – Predefined words with special meaning in SQL.
7. **Operators** – Symbols or words used for calculations or comparisons.
8. **Functions** – Built-in routines for processing values.

6. What is the general structure of an SQL SELECT statement?

The general structure of an SQL SELECT statement is:

```
SELECT column_list
FROM table_name
[WHERE condition]
[GROUP BY column]
[HAVING condition]
[ORDER BY column ASC|DESC];
```

7. What is the role of clauses in SQL statements?

Clauses define the specific actions and conditions of an SQL statement. Each clause has a particular role:

- **SELECT** → specifies the columns to retrieve.
- **FROM** → specifies the source table(s).
- **WHERE** → filters rows based on conditions.
- **GROUP BY** → groups rows for aggregation.
- **HAVING** → applies conditions on groups.
- **ORDER BY** → sorts the result set.

8. What are SQL constraints, and what is the difference between PRIMARY KEY and FOREIGN KEY, and between NOT NULL and UNIQUE?

1. **Constraints in SQL** → Rules applied to table columns to ensure accuracy and reliability of data.
    - **NOT NULL** → ensures a column cannot have NULL values.
    - **UNIQUE** → ensures all values in a column are distinct.
    - **PRIMARY KEY** → uniquely identifies each record (combines NOT NULL + UNIQUE).
    - **FOREIGN KEY** → enforces a relationship between two tables.
    - **CHECK** → ensures values satisfy a specific condition.

  - **DEFAULT** → provides a default value when no value is supplied.
2. **PRIMARY KEY vs FOREIGN KEY** →
   - **PRIMARY KEY** uniquely identifies a record in its own table.
   - **FOREIGN KEY** references the PRIMARY KEY of another table to establish a relationship.
3. **NOT NULL vs UNIQUE** →
   - **NOT NULL** ensures a column must always have a value.
   - **UNIQUE** ensures no two rows can have the same value in that column.

---

9. What are the main SQL commands under Data Definition Language (DDL), and what is the purpose of CREATE?

1. **DDL (Data Definition Language)** → SQL commands that define, alter, and remove database structures (tables, databases, schemas).
2. **CREATE command** → used to create new databases, tables, or other objects.
3. **Purpose of specifying data types & constraints** → ensures correct storage, consistency, validation, and integrity of data.

---

10. What is the use of the ALTER command, and what operations can be performed with it?

1. **Use of ALTER** → modifies an existing table structure (add, change, or remove columns and constraints).
2. **Operations with ALTER** →
   - **Add column** → ALTER TABLE ... ADD ...
   - **Modify column** → ALTER TABLE ... MODIFY ...
   - **Drop column** → ALTER TABLE ... DROP ...

---

11. What is the function of the DROP command, and what are its implications?

1. **Function** → permanently removes a database object (table, database, view, etc.).
2. **Implications** → deletes structure and data permanently, cannot be rolled back.

---

12. What are the SQL commands under Data Manipulation Language (DML), and why is the WHERE clause important in UPDATE and DELETE?

1. **INSERT** → adds new records.
   UPDATE → modifies existing records.
   DELETE → removes records.
2. **WHERE in UPDATE/DELETE** → ensures only specific rows are affected. Without WHERE, all rows may change or be deleted.

---

13. What is Data Query Language (DQL), and how are ORDER BY and WHERE used in SQL queries?

1. **SELECT** → used to query and retrieve data from tables.
2. **ORDER BY** → sorts results.
   WHERE → filters results by condition.

---

14. What are Data Control Language (DCL) commands, and how do they manage user privileges?

1. **GRANT** → gives permissions to users.
   REVOKE → removes permissions.
2. **Managing privileges** → control access (read, write, update, delete) to ensure security.

---

15. What are Transaction Control Language (TCL) commands, and how do they ensure transaction management?

1. **COMMIT** → saves all changes permanently.
   ROLLBACK → undoes changes since the last COMMIT.
2. **Transaction management** → ensures atomicity, consistency, isolation, durability (ACID properties).

---

16. What are SQL joins, and what is the difference between INNER, LEFT, RIGHT, and FULL OUTER joins?

1. **JOIN** → combines rows from multiple tables based on related columns.
   - **INNER JOIN** → returns matching rows from both tables.
   - **LEFT JOIN** → all rows from left table + matching from right.
   - **RIGHT JOIN** → all rows from right table + matching from left.
   - **FULL OUTER JOIN** → all rows from both tables, whether matching or not.
2. **Use of joins** → retrieve combined data across multiple tables.

---

17. What is the use of SQL GROUP BY, and how does it differ from ORDER BY?

1. **GROUP BY** → groups rows with the same values, often used with aggregate functions (COUNT, SUM, AVG).
2. **Difference** →
   - **GROUP BY** → groups data into sets.
   - **ORDER BY** → sorts the final results.

---

18. What is an SQL stored procedure, and how is it different from queries? What are its advantages?

1. **Stored Procedure** → precompiled SQL code stored in the database, executed with parameters.
   - Unlike queries, procedures can contain multiple SQL statements.
2. **Advantages** → improves performance, reusability, security, and reduces redundancy.

---

19. What is an SQL view, and what are its advantages?

1. **View** → a virtual table based on results of a query; does not store data physically.
2. **Advantages** → simplifies queries, enhances security, hides complexity, and ensures consistency.

---

20. What are SQL triggers, and what is the difference between INSERT, UPDATE, and DELETE triggers?

1. **Trigger** → special stored procedure executed automatically on events (INSERT, UPDATE, DELETE).
   - Types: BEFORE, AFTER, INSTEAD OF triggers.
2. **Difference** →
   - **INSERT Trigger** → fires when new rows are added.
   - **UPDATE Trigger** → fires when data is modified.
   - **DELETE Trigger** → fires when rows are deleted.

---

21. What is PL/SQL, and what are its benefits?

1. **PL/SQL** → Procedural Language extension of SQL (by Oracle). It supports loops, conditions, and modular programming.
2. **Benefits** → improves performance, supports procedural logic, reduces network traffic, and enhances security.

---

22. What are PL/SQL control structures, and why are they important?

1. **Control structures** → logical flow mechanisms.
   - **IF-THEN** → executes code conditionally.
   - **LOOP** → repeats code multiple times.
2. **Importance** → allows complex operations and decision-making inside SQL blocks.

---

23. What are SQL cursors, and what is the difference between implicit and explicit cursors? When would you use explicit cursors?

1. **Cursor** → pointer to retrieve query results row by row.
   - **Implicit cursor** → automatically created by SQL for queries.
   - **Explicit cursor** → user-defined, gives more control.
2. **Explicit cursor use** → when row-by-row processing or iteration is required.

---

24. What is the use of ROLLBACK and COMMIT with SAVEPOINT in SQL transactions?

1. **SAVEPOINT** → sets a checkpoint within a transaction.
   - **COMMIT** → makes all changes permanent, including those after savepoints.
   - **ROLLBACK** → can undo changes up to a savepoint.
2. **Usefulness** → helps undo part of a transaction without discarding the whole.