Introduction to SQL

1. What is SQL and why is it important?

SQL (Structured Query Language) is the main language used to manage and work with databases. It lets you:

- Add new records
- Get data that's already stored
- Change or remove data
- Create or edit the structure of tables, views, etc.

Why it matters:

- Makes it easier to handle databases
- Uses simple commands
- Offers control over user access
- Works with all major database systems like MySQL, Oracle, etc.

---

2. Difference Between DBMS and RDBMS:

Feature	DBMS	RDBMS
Meaning	Database Management System	Relational Database Management System
Data Storage	Stores data in files or basic formats	Stores data in table format (rows/columns)
Relationships	No relation support	Supports relationships (keys between tables)
Data Integrity	Low	High (uses rules and constraints)
Examples	MS Access, simple file-based DB	MySQL, Oracle, SQL Server, PostgreSQL

---

3. SQL's Role in Relational Databases:

SQL helps:

- Build or change table structures (CREATE, ALTER, DROP)
- Add or change data (INSERT, UPDATE)
- Read data (SELECT)
- Set rules (constraints, keys)
- Control who can do what (GRANT, REVOKE)
  It acts as a bridge between users and the database.

---

4. Key Features of SQL:

- **Query Data:** Use SELECT to read info

- **Modify Data:** INSERT, UPDATE, DELETE
- **Structure Data:** CREATE, DROP, ALTER tables
- **Control Access:** GRANT, REVOKE permissions
- **Transaction Handling:** COMMIT, ROLLBACK
- **Secure:** Manages data access properly
- **Works with Many Databases**

SQL Syntax

1. Basic Elements of SQL:

- **Keywords:** SELECT, WHERE, etc.
- **Identifiers:** Names of tables/columns
- **Operators:** =, <>, >, LIKE, etc.
- **Literals:** Fixed values like 'John', 25
- **Clauses:** SELECT, FROM, GROUP BY, etc.
- **Functions:** COUNT(), AVG(), NOW()
- **Comments:** -- for single line, /* */ for multiple lines

---

2. General Query Format:

```
sql
CopyEdit
SELECT column1, column2
FROM table_name
WHERE condition
GROUP BY column
HAVING condition
ORDER BY column ASC|DESC;
```

Example:

```
sql
CopyEdit
SELECT name, age FROM students WHERE age > 18 ORDER BY age DESC;
```

---

3. Use of Clauses in SQL:

- **SELECT:** Pick columns
- **FROM:** Choose table
- **WHERE:** Filter rows
- **GROUP BY:** Group data by a column
- **HAVING:** Filter after grouping
- **ORDER BY:** Sort results

SQL Constraints

1. What are constraints?
Rules applied to columns to keep data valid and accurate:

- **NOT NULL** – Can't be empty
- **UNIQUE** – All values must be different
- **PRIMARY KEY** – Unique and not null
- **FOREIGN KEY** – Links to another table
- **CHECK** – Must meet a condition
- **DEFAULT** – Auto-fill if no value given

---

2. Primary Key vs Foreign Key:

- **Primary Key:** Uniquely identifies each row; must be unique and not null
- **Foreign Key:** Connects to primary key of another table; can be null or repeated

---

3. NOT NULL vs UNIQUE:

- **NOT NULL:** Value must be present
- **UNIQUE:** All values must be different
  Together, they make data reliable.

DDL (Data Definition Language)

1. What is DDL?
Commands to define or change database objects:

- **CREATE** – Make a new table or DB
- **ALTER** – Change existing table
- **DROP** – Delete table or DB
- **TRUNCATE** – Remove all data quickly

---

2. CREATE Command Syntax:

```
sql
CopyEdit
CREATE TABLE table_name (
  column1 datatype constraint,
  column2 datatype constraint
);
```

---

3. Why Use Data Types and Constraints:

- **Data Types:** Decide what kind of data a column can store
- **Constraints:** Keep the data valid (e.g., no empty values or duplicates)

ALTER Command

1. What does ALTER do?
Used to make changes to a table:

- Add new columns
- Edit existing columns
- Remove columns
- Rename table or columns
- Add/remove constraints

2. Examples:

```
sql
CopyEdit
ALTER TABLE students ADD email VARCHAR(50);
ALTER TABLE students MODIFY name VARCHAR(100);
ALTER TABLE students DROP COLUMN age;
```

DROP Command

1. What does DROP do?
Permanently deletes database items like tables or whole databases.

2. Effects:

- Table and data are gone forever
- Constraints and indexes are removed
- Related apps or queries may break

DML (Data Manipulation Language)

1. INSERT, UPDATE, DELETE:

- **INSERT:** Add new row
- **UPDATE:** Change existing data
- **DELETE:** Remove rows

2. Why use WHERE in UPDATE/DELETE?

- It controls **which rows** get changed or removed
- Without WHERE: all rows may be affected!

DQL (Data Query Language)

1. SELECT Statement:
Used to get data from a table.
Example:

```
sql
CopyEdit
SELECT * FROM students;
```

2. WHERE & ORDER BY:

- **WHERE:** Filters rows based on a condition
- **ORDER BY:** Sorts results by one or more columns

DCL (Data Control Language)

1. GRANT and REVOKE:

- **GRANT:** Gives access to users
- **REVOKE:** Takes access back

2. How to Manage Access:
Example:

```
sql
CopyEdit
GRANT SELECT ON students TO user1;
REVOKE SELECT ON students FROM user1;
```

TCL (Transaction Control Language)

1. COMMIT and ROLLBACK:

- **COMMIT:** Save all changes
- **ROLLBACK:** Undo changes

2. Transactions:
A group of SQL operations that are handled together.
Follows ACID principles:

- Atomicity, Consistency, Isolation, Durability

Joins in SQL

1. What is a JOIN?
Combines rows from two tables using a related column.

Types:

- **INNER JOIN:** Only matching rows
- **LEFT JOIN:** All from left table + matches
- **RIGHT JOIN:** All from right table + matches
- **FULL OUTER JOIN:** All rows from both tables

2. Use of JOINs:
Used to fetch related data stored across multiple tables.

GROUP BY Clause

1. What it does:
Groups rows with the same values and lets you use functions like COUNT(), AVG(), etc.

```
sql
CopyEdit
SELECT department, AVG(salary) FROM employees GROUP BY department;
```

2. GROUP BY vs ORDER BY:

- **GROUP BY:** Groups rows
- **ORDER BY:** Sorts results
  GROUP BY comes before ORDER BY in queries.

Stored Procedures

1. What is a Stored Procedure?
A stored procedure is a saved block of SQL code that can run many times. It can include logic, conditions, loops, and parameters.

Example:

```
sql
CopyEdit
CREATE PROCEDURE GetEmployeeByDept (@DeptName VARCHAR(50))
AS
BEGIN
  SELECT * FROM employees WHERE department = @DeptName;
END;
```

2. Benefits:

- Saves time (reusable)
- Faster (precompiled)
- Safer (limited access)
- Centralized logic
- Less network traffic

SQL Views

1. What is a View?
A view is a saved SQL query that looks like a table but doesn't store data.

Example:

```
sql
CopyEdit
CREATE VIEW HighSalaryEmployees AS
SELECT name, salary FROM employees WHERE salary > 50000;
```

2. Advantages:

- Hides complexity
- Adds security
- Can be reused
- Allows logical data independence

SQL Triggers

1. What is a Trigger?
A trigger runs automatically when something happens (INSERT, UPDATE, DELETE) on a table.

Types:

- **BEFORE Trigger:** Runs before the action
- **AFTER Trigger:** Runs after
- **INSTEAD OF Trigger:** Replaces the action

2. Types Based on Events:

Trigger Type	When It Runs	Example Use Case
INSERT	After inserting a row	Log new user data
UPDATE	After modifying a row	Track salary changes
DELETE	After deleting a row	Store deleted data in backup table

---

PL/SQL Introduction

1. What is PL/SQL?
Oracle’s extension of SQL with added features like loops, IF/ELSE, variables, and error handling.

2. Benefits:

- Logic + SQL in one
- Better performance
- Easier to reuse code
- Built-in error handling
- Safe and secure
- Works closely with SQL

PL/SQL Control Structures

1. What Are They?
Used for flow control in PL/SQL.

- **IF...THEN:** Run code if a condition is true
- **LOOP:** Repeat code until condition met

Examples:

```
plsql
CopyEdit
IF salary > 50000 THEN
  bonus := 1000;
END IF;

i := 1;
LOOP
  DBMS_OUTPUT.PUT_LINE(i);
  i := i + 1;
  EXIT WHEN i > 5;
END LOOP;
```

PL/SQL Cursors

1. What is a Cursor?
Used to handle multiple rows from a query, one by one.

Types:

- **Implicit Cursor:** Auto-created for simple queries
- **Explicit Cursor:** Manually handled for multi-row queries

2. When to Use Explicit Cursors:

- Need custom row-by-row processing
- Handle complex logic
- When SELECT returns multiple rows

Savepoints in Transactions

1. What is a SAVEPOINT?

A marker inside a transaction to which you can roll back if needed.

```
sql
CopyEdit
SAVEPOINT sp1;
ROLLBACK TO sp1;
```

2. When to Use:

- Undo part of a transaction
- Handle errors safely
- Keep stable steps while testing logic