DBMS Core Concepts + SQL Command Categories

1. DBMS (Database Management System) Basics

- A DBMS is software used to store, retrieve, and manage data efficiently.
- It supports data abstraction, security, concurrency, and consistency.

Key Components:
- Data: Stored information
- DBMS Engine: Core service that processes queries
- Database Schema: Structure of tables
- Query Processor: Interprets SQL queries
- Transaction Manager: Ensures atomicity and isolation
- Storage Manager: Handles data storage and indexing

2. Types of SQL Statements

SQL commands are grouped into 5 major categories:

A. DQL (Data Query Language)
- SELECT: Retrieves data from a table
  Example:
```
SELECT * FROM employees;
SELECT name, salary FROM employees WHERE department = 'IT';
```

B. DML (Data Manipulation Language)
- INSERT: Adds new rows
- UPDATE: Modifies existing rows
- DELETE: Removes rows
  Examples:
```
INSERT INTO employees (name, dept) VALUES ('John', 'HR');
UPDATE employees SET salary = 60000 WHERE id = 1;
DELETE FROM employees WHERE id = 5;
```

C. DDL (Data Definition Language)
- CREATE: Creates tables and database objects
- ALTER: Modifies structure
- DROP: Deletes objects
- TRUNCATE: Removes all rows from a table quickly
  Examples:
```
CREATE TABLE employees (id INT, name VARCHAR(50));
ALTER TABLE employees ADD COLUMN salary INT;
DROP TABLE employees;
```

D. TCL (Transaction Control Language)
- COMMIT: Saves changes
- ROLLBACK: Undoes changes
- SAVEPOINT: Sets a checkpoint in transaction
- SET TRANSACTION: Sets properties for a transaction
  Example:
```
BEGIN;
UPDATE accounts SET balance = balance - 100 WHERE id = 1;
SAVEPOINT save1;
```

```
UPDATE accounts SET balance = balance + 100 WHERE id = 2;
ROLLBACK TO save1;
COMMIT;
```

E. DCL (Data Control Language)
- GRANT: Gives access rights
- REVOKE: Removes access rights

Example:

```
GRANT SELECT ON employees TO user1;
REVOKE SELECT ON employees FROM user1;
```

3. Core DBMS Concepts

Entity & Attributes
- Entity: Real-world object (e.g., Student)
- Attributes: Properties of an entity (e.g., Name, Roll No)

Schema & Instance
- Schema: Structure of a database (design)
- Instance: Actual data in the database at a point in time

Keys
- Primary Key: Unique identifier
- Foreign Key: Reference to a primary key in another table
- Candidate Key: Possible unique identifiers
- Composite Key: Combination of fields as a key

Constraints
- NOT NULL: Column cannot be empty
- UNIQUE: No duplicate values
- CHECK: Enforces condition
- DEFAULT: Default value if not provided
- FOREIGN KEY: Ensures referential integrity

Normalization
1NF Atomic values
2NF No partial dependency
3NF No transitive dependency
BCNF Every determinant is a candidate key

Joins
- INNER JOIN: Returns matching rows
- LEFT JOIN: All rows from left + matched rows
- RIGHT JOIN: All rows from right + matched rows
- FULL JOIN: All rows when there is a match
- SELF JOIN: Joins table with itself

ACID Properties (Transactions)
- Atomicity: All operations succeed or none
- Consistency: Maintains database validity
- Isolation: Transactions dont interfere
- Durability: Changes persist after commit

Indexing

- Used to speed up data retrieval
  Example: CREATE INDEX idx_name ON employees(name);

Views
- Virtual table based on result of SQL query
  Example:
    CREATE VIEW hr_employees AS
    SELECT * FROM employees WHERE dept = 'HR';

Stored Procedures & Triggers
- Stored Procedure: SQL block stored and executed by name
  Example:
    CREATE PROCEDURE update_salary()
    BEGIN
      UPDATE employees SET salary = salary + 1000 WHERE dept = 'Sales';
    END;

- Trigger: Auto-executes on INSERT/UPDATE/DELETE
  Example:
    CREATE TRIGGER before_insert
    BEFORE INSERT ON employees
    FOR EACH ROW
    SET NEW.salary = 30000;