DBMS SQL Viva Questions (Basic to Moderate)

DBMS – Basic Questions

1. **What is DBMS?**
   A software system that stores, manages and retrieves data efficiently.

2. **What is a database?**
   A structured collection of related data.

3. **What is a table?**
   A table stores data in rows and columns.

4. **What is a primary key?**
   A unique identifier for each record; cannot be NULL.

5. **What is a foreign key?**
   A key that references primary key of another table.

6. **What is a candidate key?**
   All possible attributes that can act as a primary key.

7. **What is a super key?**
   A set of attributes that uniquely identifies a record.

8. **What is a composite key?**
   A key made of two or more attributes.

9. **What is normalization?**
   Process of organizing data to reduce redundancy.

10. **Explain 1NF, 2NF, 3NF.**
    1NF: No repeating groups.
    2NF: No partial dependency.
    3NF: No transitive dependency.

11. **What is a view?**
    Virtual table created using a SELECT query.

12. **What is a transaction?**
    A set of operations performed as a single logical unit.

13. **Explain ACID properties.**
    Atomicity, Consistency, Isolation, Durability.

14. **What is SQL?**
    Structured Query Language used for database operations.

SQL – Basic to Moderate Questions

1. **Types of SQL commands?**
   DDL, DML, DCL, TCL, DQL.

2. **What is DDL? Give examples.**
   DDL stands for Data Definition Language.
   Examples: CREATE, ALTER, DROP.

3. **What is DML?**
   DML stands for Data Manipulation Language.
   Examples: INSERT, UPDATE, DELETE.

4. **Difference between DDL and DML?**
   DDL modifies structure of tables; auto-committed.
   DML modifies data inside tables; not auto-committed.

5. **Difference between CHAR and VARCHAR?**
   CHAR stores fixed-length strings; faster but wastes space.
   VARCHAR stores variable-length strings; saves space.

6. **Difference: DELETE vs TRUNCATE?**
   DELETE removes rows one by one; TRUNCATE removes all rows instantly.

7. **What is a constraint?**
   Rule applied on a column (PRIMARY KEY, UNIQUE, CHECK, NOT NULL).

8. **How to create a table?**
   ```
   CREATE TABLE student(id INT PRIMARY KEY, name VARCHAR(20));
   ```

9. **How to insert data?**
   ```
   INSERT INTO student VALUES(1,'Aman');
   ```

10. **How to update a row?**
    `UPDATE student SET name='Ravi' WHERE id=1;`

11. **How to delete a row?**
    `DELETE FROM student WHERE id=1;`

12. **What is SELECT query?**
    Retrieves data from table.

13. **How to select all rows?**
    `SELECT * FROM student;`

14. **How to use WHERE clause?**
    `SELECT * FROM student WHERE id=2;`

15. **What is LIKE operator?**
    Pattern matching: `SELECT * FROM emp WHERE name LIKE 'A%';`

16. **Difference between WHERE and HAVING?**
    WHERE filters rows; HAVING filters groups.

17. **What are aggregate functions?**
    COUNT, SUM, AVG, MIN, MAX.

18. **Example of GROUP BY?**
    `SELECT dept, COUNT(*) FROM emp GROUP BY dept;`

19. **What is ORDER BY?**
    Sorts result in ASC/DESC order.

20. **Explain JOINs.**
    Join combines rows from two tables based on related columns.

21. **Types of joins?**
    INNER, LEFT, RIGHT, FULL.

22. **Example of INNER JOIN?**
    `SELECT s.name, m.marks FROM student s JOIN marks m ON s.id=m.id;`

23. **What is a subquery?**
    A query inside another query.

24. **Example of subquery?**
    `SELECT name FROM emp WHERE salary > (SELECT AVG(salary) FROM emp);`

25. **What is DISTINCT?**
    Removes duplicate values.

26. **What is LIMIT?**
    Restricts number of rows returned. Example: `LIMIT 5;`

27. **What is ALTER used for?**
    Modify structure of table (add/modify/drop column).

28. **What is DROP?**
    Removes table completely from database.

29. **What is COMMIT?**
    Saves transaction permanently.

30. **What is ROLLBACK?**
    Undo a transaction.

31. **What is UNION?**
    Combines result of two queries and removes duplicates.

32. **What is UNION ALL?**
    Same as UNION but keeps duplicates.

SQL Practical Notes (From Notebook)

1. **show databases;** → shows created databases.

2. **create database demo;** → creates database.

3. **use demo;** → switches to database.

4. **create table student(**
   name VARCHAR(30),
   id INT NOT NULL PRIMARY KEY,
   address VARCHAR(50),
   marks INT);
   → creates student table.

5. **desc student;** → describes table structure.

6. **insert into student(marks, id, name, address) values (78,12,'Kailash','Kanpur');**

7. **insert into student values ('Kailash', 23, 'Varanasi', 16);**

8. **insert into student values ('Ravi', 45, 'Kondli', 79), ('Roni', 17, 'Delhi', 90);**

9. **select * from student;** → view whole table.

10. **select * from student where id = 12;** → specific row.

11. **update student set address='Bhubaneshwar' where id=45;** → update row.
12. **alter table student add phoneNo int;** → add column.
13. **alter table student modify column name varchar(60);** → change datatype/size.
14. **alter table student drop column phoneNo;** → drop column. [DDL]
15. **delete from student where name='Kamal';** → delete row. [DML]
16. **select sum(marks) from student;** → aggregate functions: SUM, AVG, COUNT, MAX, MIN.
17. **select * from student order by id;** → ascending order.
18. **select * from student order by id desc;** → descending order.
19. **select * from student where name like 'R%';** → names starting with R.
20. **select * from student where name like '_a%';** → 2nd letter = 'a'.
21. **select * from student inner join employee**
    on student.id = employee.id;

    LEFT JOIN (simple explanation):
    Left table is the main table.
    Shows all rows from the LEFT table and only matching rows from the right table.
    If there is no match, the right-side values become NULL.
    Example: All students appear, even if some have not paid fees (fee = NULL).

    RIGHT JOIN (simple explanation):
    Right table is the main table.
    Shows all rows from the RIGHT table and only matching rows from the left table.
    If there is no match, the left-side values become NULL.
    Example: All departments appear, even if they have no employees (employee = NULL).

    Memory Trick:
    LEFT JOIN → Left table never loses rows.
    RIGHT JOIN → Right table never loses rows.

    CROSS JOIN → every row × every row.