Sql Queries

✅ 1. Database Queries

▸ Database बनाना

```
CREATE DATABASE company_db;
```

▸ Database देखना

```
SELECT datname FROM pg_database;
```

▸ Database delete

```
DROP DATABASE company_db;
```

✅ 2. Table Queries (DDL)

▸ Table बनाना

```
CREATE TABLE employee (
  id INT,
  name VARCHAR(50),
  salary INT,
  dept VARCHAR(30)
);
```

▸ सभी tables देखना

```
SELECT table_name
FROM information_schema.tables
WHERE table_schema='public';
```

▸ Table delete

```
DROP TABLE employee;
```

✅ 3. Column / Structure देखना

```
SELECT column_name, data_type
```

```
FROM information_schema.columns
WHERE table_name = 'employee';
```

✅ 4. Insert Queries

▸ Single row insert

```
INSERT INTO employee
VALUES (1,'Ravi',30000,'IT');
```

▸ Selected columns

```
INSERT INTO employee(id,name)
VALUES (2,'Amit');
```

✅ 5. Select Queries (Very Important)

▸ पूरा data

```
SELECT * FROM employee;
```

▸ Specific columns

```
SELECT name, salary FROM employee;
```

▸ WHERE

```
SELECT * FROM employee
WHERE dept='IT';
```

▸ AND / OR

```
SELECT * FROM employee
WHERE dept='IT' AND salary > 20000;
```

✅ 6. Update Queries

```
UPDATE employee
SET salary = 40000
WHERE id = 1;
```

✅ 7. Delete Queries

```
DELETE FROM employee
WHERE id = 2;
```

✅ 8. ORDER BY

```
SELECT * FROM employee
ORDER BY salary DESC;
```

✅ 9. LIMIT

```
SELECT * FROM employee
LIMIT 5;
```

✅ 10. Aggregate Functions

▸ COUNT

```
SELECT COUNT(*) FROM employee;
```

▸ SUM

```
SELECT SUM(salary) FROM employee;
```

▸ AVG

```
SELECT AVG(salary) FROM employee;
```

▸ MAX / MIN

```
SELECT MAX(salary), MIN(salary)
FROM employee;
```

✅ 11. GROUP BY

```
SELECT dept, COUNT(*)
FROM employee
GROUP BY dept;
```

✅ 12. HAVING

```
SELECT dept, COUNT(*)
FROM employee
GROUP BY dept
HAVING COUNT(*) > 2;
```

✅ 13. IN

```
SELECT * FROM employee
WHERE dept IN ('IT','HR');
```

✅ 14. BETWEEN

```
SELECT * FROM employee
WHERE salary BETWEEN 20000 AND 40000;
```

✅ 15. LIKE

```
SELECT * FROM employee
WHERE name LIKE 'R%';
```

✅ 16. DISTINCT

```
SELECT DISTINCT dept
FROM employee;
```

✅ 17. JOIN Queries

मान लो दूसरी table है department

department(dept_id, dept_name)

▸ INNER JOIN

```
SELECT e.name, d.dept_name
FROM employee e
INNER JOIN department d
ON e.dept = d.dept_name;
```

▸ LEFT JOIN

```
SELECT *
FROM employee e
LEFT JOIN department d
ON e.dept = d.dept_name;
```

▸ RIGHT JOIN

```
SELECT *
FROM employee e
RIGHT JOIN department d
ON e.dept = d.dept_name;
```

✅ 18. Subquery

```
SELECT *
FROM employee
WHERE salary >
(
  SELECT AVG(salary)
  FROM employee
);
```

✅ 19. EXISTS

```
SELECT *
FROM employee e
WHERE EXISTS
(
 SELECT 1
 FROM department d
 WHERE d.dept_name = e.dept
);
```

✅ 20. Constraints

▸ Primary Key

```
CREATE TABLE emp2(
id INT PRIMARY KEY,
name VARCHAR(50)
);
```

▸ NOT NULL

```
name VARCHAR(50) NOT NULL
```

▸ UNIQUE

```
email VARCHAR(100) UNIQUE
```

✅ 21. ALTER TABLE

▸ Column add

```
ALTER TABLE employee
ADD email VARCHAR(100);
```

▸ Column drop

```
ALTER TABLE employee
DROP COLUMN email;
```

✅ 22. View

```
CREATE VIEW emp_view AS
SELECT name, salary
FROM employee;
SELECT * FROM emp_view;
```

✅ 23. Index

```
CREATE INDEX idx_emp_name
ON employee(name);
```

✅ 24. Transaction

```
BEGIN;

UPDATE employee
SET salary = salary + 5000
WHERE dept='IT';

COMMIT;
```

✅ 25. Rollback

```
BEGIN;

DELETE FROM employee;

ROLLBACK;
```

✅ 26. CASE (Interview में बहुत पूछा जाता है)

```
SELECT name,
CASE
  WHEN salary >= 40000 THEN 'High'
```

```
  WHEN salary >= 25000 THEN 'Medium'
  ELSE 'Low'
END AS salary_level
FROM employee;
```

✅ 27. NULL handling

```
SELECT *
FROM employee
WHERE salary IS NULL;
```

✅ 28. COALESCE

```
SELECT name,
COALESCE(salary,0)
FROM employee;
```