🟢 BASIC LEVEL

🔷 DDL (Create/Modify Tables)

1. **Create a Table**
   Write a query to create a table named employees with columns:
   - emp_id (INT, Primary Key)
   - first_name (VARCHAR(50))
   - last_name (VARCHAR(50))
   - hire_date (DATE)
2. **Add a New Column**
   Add a column salary (DECIMAL(10,2)) to the existing employees table.
3. **Drop a Table**
   Delete the table old_backups from the database if it exists.
4. **Rename a Table**
   Rename table customers to clients.

🔷 DML (Insert/Update/Delete Data)

5. **Insert a Row**
   Insert a department with ID 101 and name 'Marketing' into the departments table.
6. **Update Records**
   Increase salary by 10% for all employees in department 101.
7. **Delete Records**
   Delete all products from products table where discontinued = 1.

🔷 JOINs (Combining Tables)

8. **INNER JOIN**
   Show employee names and their department names using employees and departments tables.
9. **LEFT JOIN**
   Show all departments and the number of employees in each department (include departments with zero employees).
10. **Self JOIN**
    Show each employee and their manager name from the same employees table (manager_id references emp_id).

🟡 INTERMEDIATE LEVEL

🔷 DDL

11. **Alter Column & Add Constraint**
    Change salary column to NUMERIC(12,2) and ensure all salaries are greater than 0 using a CHECK constraint.
12. **Add Foreign Key**
    In employees, make dept_id reference departments(dept_id) with ON DELETE SET NULL.
13. **Create Index**
    Create an index on orders(order_date) to speed up search by date.

🔷 DML

14. **Insert with SELECT (Upsert Style)**
    Insert a product into products, or update its price if it already exists (UPSERT logic).
15. **Update with JOIN**
    Increase salaries by 5% for employees who work in departments located in 'Delhi'.
16. **Delete Using EXISTS**
    Delete all customers from customers table who have never placed an order (check in orders table).

🔷 JOINs

17. **RIGHT JOIN**
    Show all suppliers and the number of products they provide (even if it's 0).
18. **FULL OUTER JOIN**
    List all orders and all shipments. Match them using order_id, and show unmatched data from both sides.
19. **CROSS JOIN**
    Show all possible combinations of top 5 products and top 3 discount codes.

🔴 ADVANCED LEVEL

🔷 DDL

20. **Partitioned Table**
    Write SQL to create a table sales_2025 partitioned by quarter using sale_date.
21. **Create Sequence and Use It**
    Create a sequence invoice_seq starting from 1000. Use it to auto-generate invoice_id in invoices table.

22. **Create a View**
    Create a view v_active_employees showing only employees whose status = 'ACTIVE'.

🔷 DML

23. **Common Table Expression (CTE) Update**
    Write a CTE to calculate average salary and then update salaries for those earning below average.
24. **Delete with RETURNING**
    Delete orders older than 5 years and return the order_id and customer_id for deleted rows.
25. **Merge (Upsert)**
    Merge products_new into products – if the product exists, update it, else insert it.

🔷 JOINs

26. **Join with Aggregation**
    List top 3 highest-paid employees in each department along with their manager names.
27. **Subquery Join (Correlated)**
    Show products with their last order date using a correlated subquery and join.
28. **Recursive JOIN**
    Using a hierarchical categories table (with parent_id), find the level/depth of each category.
29. **Anti-Join (NOT EXISTS)**
    Show products that have never been reordered after their first sale.
30. **Pattern Matching JOIN (Optional if supported)**
    Find customers who placed 3 increasing-value orders on 3 consecutive days.

✅ Bonus – Transaction & Error Handling

31. **Transactional Query**
    Write a script that:

- Backs up data from employees to employees_backup
- Deletes employees hired before 2000
- If backup or delete fails, rollback the transaction