1. To Implement the DDL Commands in SQL

1. Create a table named **Employees** with the columns `EmpID`, `Name`, `Position`, `Salary`, and `HireDate`.
2. Add a new column `Department` to the **Employees** table.
3. Modify the data type of the `Salary` column in **Employees** to `DECIMAL(10,2)`.
4. Drop the **Employees** table.
5. Create a table **Library** with a composite primary key on `BookID` and `AuthorID`.

---

2. To Implement the DML Commands in SQL

6. Insert five records into the **Employees** table.
7. Update the `Salary` of employees in the **Employees** table who have the position "Manager" to `70000`.
8. Delete records from the **Employees** table where the `Position` is "Intern".
9. Retrieve the names and salaries of all employees earning more than `50000`.
10. Copy all data from **Employees** to a new table named **Archived Employees**.

---

3. To Implement the DCL & TCL Commands in SQL

11. Grant `SELECT` and `UPDATE` permissions on the **Employees** table to a user named `AdminUser`.
12. Revoke the `UPDATE` permission from `AdminUser`.
13. Create a transaction to insert three records into **Orders** and rollback if any insert fails.
14. Use `SAVEPOINT` to create a point during a transaction and roll back to that point after updating **Employees**.
15. Commit a transaction after updating the `Position` of employees in **Employees** to "Senior Developer".

---

4. To Write a Query to Implement Constraints in SQL

16. Create a table **Students** with a `StudentID` as the primary key.
17. Create a table **Courses** with a foreign key referencing `StudentID` in **Students**.
18. Add a `NOT NULL` constraint to the `Email` column in **Students**.
19. Create a table **Products** with a check constraint that ensures `Price > 0`.
20. Add a `UNIQUE` constraint on the `PhoneNumber` column in the **Customers** table.

---

5. To Implement Aggregate Functions Using SQL Queries

21. Calculate the average salary of all employees in the **Employees** table.
22. Find the maximum price of products in the **Products** table.
23. Count the total number of employees in the **Employees** table.
24. Calculate the total sales amount from the **Sales** table.
25. Find the minimum age of students from the **Students** table.

---

6. To Study and Implement SET Operations in SQL

26. Write a query using `UNION` to combine the results of two tables: **Orders2023** and **Orders2024**.
27. Write a query using `INTERSECT` to find common products in two tables: **WarehouseA** and **WarehouseB**.
28. Use `EXCEPT` to display employees present in **DepartmentA** but not in **DepartmentB**.
29. Combine two tables using `UNION ALL` and display the results.
30. Write a query to use a set operation with a `WHERE` clause on both tables.

---

7. To Study BETWEEN & IN Operators in SQL

31. Retrieve all employees from the **Employees** table where `Salary` is between `30000` and `70000`.
32. Retrieve all orders from the **Orders** table where the `OrderDate` is between '2023-01-01' and '2023-12-31'.
33. Find all students whose `Course` is in ('Math', 'Physics', 'Chemistry').
34. Retrieve products where the `Price` is between `100` and `500`.
35. Retrieve records from **Customers** where the `City` is in ('New York', 'Los Angeles', 'Chicago').

---

8. To Study and Execute ORDER BY, GROUP BY, and HAVING Clause in SQL

36. Retrieve all employees from the **Employees** table ordered by `Salary` in descending order.
37. Group the employees by their `Department` and calculate the total salary for each department.
38. Retrieve all products grouped by `Category` with a condition that the average price is greater than `50`.
39. Display the total number of orders placed by each customer from the **Orders** table.
40. Retrieve all customers grouped by `City` and filtered by a `HAVING` clause where the count of customers in the city is greater than `10`.

---

9. Write SQL Queries to Implement JOINS

41. Use an `INNER JOIN` to retrieve all orders along with the customer details from **Orders** and **Customers** tables.
42. Use a `LEFT JOIN` to find all customers who have not placed any orders.
43. Use a `RIGHT JOIN` to find all orders, including those with customers not present in the **Customers** table.
44. Use a `FULL OUTER JOIN` to combine the data from **Products** and **Suppliers**.
45. Use a `SELF JOIN` to find employees in the same department as another employee in the **Employees** table.

---

10. Write SQL Queries to Implement Operators in SQL

46. Write a query using the `LIKE` operator to find all customers whose names start with 'J'.
47. Use the `AND` and `OR` operators to retrieve products with a `Price` less than `50` or more than `200`.
48. Use the `IS NULL` operator to find all records in **Orders** where the `ShippingDate` is null.
49. Use the `NOT` operator to retrieve all employees who do not work in the "HR" department.
50. Write a query using the `%` (modulus) operator to find even `ProductID` values from the **Products** table.

Aim

Theory : write atleast 2 page theory related to topic.

Program: write query properly.

Output: write output with proper tabular format.

Result: