```
<------------------------------------Table Creation Query (DDL
COMMANDS)--------------------------->

1.Write a SQL statement to create a table named jobs including columns job_id,
job_title, min_salary, max_salary and check whether the max_salary amount
exceeding the upper limit 25000.
    :-  CREATE TABLE jobs (
        job_id  VARCHAR(10) PRIMARY KEY,
        job_title VARCHAR(50) NOT NULL,
        min_salary NUMERIC(10,2) ,
        max_salary NUMERIC(10,2) CHECK (max_salary <= 25000)
        );

2.Write a SQL statement to create a table named countries including columns
country_id, country_name and region_id and make sure that no countries except
Italy, India and China will be entered in the table.
    :-  CREATE TABLE countries(
        country_id VARCHAR(5) PRIMARY KEY,
        country_name VARCHAR(15),
        region_id VARCHAR(5)
        CHECK (country_name IN ('Italy', 'India', 'China'))
    );

3.Write a SQL statement to create a table named countries including columns
country_id,country_name and region_id and make sure that no duplicate data
against column country_id will be allowed at the time of insertion.
    :-  CREATE TABLE countries(
        country_id VARCHAR(5) PRIMARY KEY,
        country_name VARCHAR(15),
        region_id VARCHAR(5)
        UNIQUE (country_id)
    );

4.Write a SQL statement to create a table named jobs including columns job_id,
job_title, min_salary and max_salary, and make sure that, the default value for
job_title is blank and min_salary is 8000 and max_salary is NULL will be entered
automatically at the time of insertion if no value assigned for the specified
columns.
    :-  CREATE TABLE jobs (
        job_id  VARCHAR(10) PRIMARY KEY,
        job_title VARCHAR(50) DEFAULT '',
        min_salary NUMERIC(10,2) DEFAULT 8000,
        max_salary NUMERIC(10,2)
        );


5.Write a SQL statement to create a table named countries including columns
country_id, country_name and region_id and make sure that the country_id column
will be a key field which will not contain any duplicate data at the time of
insertion.
    :-  CREATE TABLE countries(
        country_id VARCHAR(5) PRIMARY KEY,
        country_name VARCHAR(15),
        region_id VARCHAR(5)
        );


6.Write a SQL statement to create a table countries including columns
country_id, country_name and region_id and make sure that the column country_id
will be unique and store an auto-incremented value.
    :-  CREATE TABLE countries(
        country_id SERIAL PRIMARY KEY,
        country_name VARCHAR(15),
        region_id VARCHAR(5)
```

```
       );
```

7.Write a SQL statement to create a table countries including columns country_id, country_name and region_id and make sure that the combination of columns country_id and region_id will be unique.

```
    :-  CREATE TABLE countries(
        country_id VARCHAR(5) PRIMARY KEY,
        country_name VARCHAR(15) NOT NULL ,
        region_id VARCHAR(5) NOT NULL,
        UNIQUE (country_id,region_id)
        );
```

8.Write a SQL statement to create a table job_history including columns employee_id, start_date, end_date, job_id and department_id and make sure that, the employee_id column does not contain any duplicate value at the time of insertion and the foreign key column job_id contain only those values which exist in the jobs table.

```
    :-  CREATE TABLE job_history (
        employee_id INT NOT NULL,
        start_date DATE NOT NULL,
        end_date DATE NOT NULL,
        job_id VARCHAR(10) NOT NULL,
        department_id INT,
        PRIMARY KEY (employee_id, start_date),
        FOREIGN KEY (job_id) REFERENCES jobs(job_id)
        );
```

<-----------------------------------DML
COMMANDS-->

1.Write a query to find the number of jobs available in the employees table.

```
    :-  SELECT COUNT(DISTINCT job_title) AS number_of_jobs
        FROM employee;
```

2.Write a query to get the total salaries payable to employees.

```
    :-SELECT SUM(salary) AS total_salaries_payable
      FROM employee;
```

3. Write a query to get the minimum salary from employees table.

```
    :-SELECT MIN(salary) AS min_salary
      FROM employee;
```

4. Write a query to get the maximum salary of an employee working as a Programmer.

```
    :-SELECT MAX(salary) AS max_salary_of_programmer
      FROM employee  WHERE job_id='IT_PROG';
```

5. Write a query to get the average salary and number of employees working in the department which ID is 90.

```
    :-SELECT AVG(salary) AS avg_salary, COUNT(*) AS num_employees
      FROM employee WHERE department_id = 90;
```

6. Write a query to get the highest, lowest, total, and average salary of all employees.

```
    :-SELECT AVG(salary) AS avg_salary,
      MAX(salary) AS highest_salary,
      MIN(salary) AS lowest_salary,
      SUM(salary) AS Total FROM employee;
```

7. Write a query to get the number of employees working in each post.

```
    :-SELECT job_id, COUNT(*) AS num_employees
```

```
      FROM employee GROUP BY job_id;

8. Write a query to get the difference between the highest and lowest salaries.
    :-SELECT MAX(salary) - MIN(salary) AS salary_difference FROM employee;

9. Write a query to find the manager ID and the salary of the lowest-paid
employee under that manager.
    :-SELECT manager_id, MIN(salary)
      FROM employee WHERE manager_id IS NOT NULL
      GROUP BY manager_id ORDER BY MIN(salary) DESC;

10. Write a query to get the department ID and the total salary payable in each
department.
    :- SELECT department_id, SUM(salary) AS total_salary_payable
       FROM employee  GROUP BY department_id;

11. Write a query to get the average salary for each post excluding programmer.
    :-SELECT job_id, AVG(salary) AS avg_salary FROM employees
      WHERE job_id <> 'Programmer' GROUP BY job_id;

12. Write a query to get the total salary, maximum, minimum and average salary
of all posts for those
    departments which ID 90.
    :-SELECT AVG(salary) AS avg_salary,
      MAX(salary) AS highest_salary,
      MIN(salary) AS lowest_salary,
      SUM(salary) AS Total FROM employee WHERE department_id= 90;

13. Write a query to get the job ID and maximum salary of each post for maximum
salary is at or above $4000.
    :-SELECT job_id, MAX(salary) AS max_salary
      FROM employees GROUP BY job_id
      HAVING MAX(salary) >= 4000;


1. Write a query to display the name, including first_name and last_name and
salary for all employees whose salary is out of the range between $10,000 and
$15,000.
    :-SELECT first_name, last_name, salary
      FROM employee
      WHERE salary NOT BETWEEN 10000 AND 15000;

2. Write a query to display the name, including first_name and last_name, and
department ID who works in the department 30 or 100 and arrange the result in
ascending order according to the department ID.
    :-SELECT first_name, last_name, department_id FROM employee
      WHERE department_id IN (30, 100)
      ORDER BY  department_id  ASC;

3. Write a query to display the name, including first_name and last_name, and
salary who works in the department either 30 or 100 and salary is out of the
range between $10,000 and $15,000.
    :-SELECT first_name, last_name, department_id FROM employee
      WHERE department_id IN (30, 100)
      AND salary NOT BETWEEN 10000 AND 15000;

4. Write a query to display the name, including first_name and last_name and
hire date for all employees who were hired in 1987.
    :-SELECT first_name, last_name, hire_date FROM employee
      WHERE hire_date BETWEEN '1987-01-01' AND '1987-12-31';

5. Write a query to get the first name of the employee who holds the letter 'c'
and 'e' in the first name.
```

```
    :-SELECT first_name FROM employee
      WHERE first_name LIKE '%c%' AND first_name LIKE '%e%';
```

6. Write a query to display the last name, job, and salary for all those employees who hasn't worked as a Programmer or a Shipping Clerk, and not drawing the salary $4,500, $10,000, or $15,000.

```
    :-SELECT last_name, job, salary FROM employee
      WHERE job NOT IN ('Programmer', 'Shipping Clerk')
      AND salary NOT IN (4500, 10000, 15000);
```

7. Write a query to display the last names of employees whose name contain exactly six characters.

```
    :-SELECT last_name FROM employee WHERE last_name LIKE '______';
```

8. Write a query to display the last name of employees having 'e' as the third character.

```
    :-SELECT last_name FROM employee WHERE last_name LIKE '__e%';
```

9. Write a query to display the jobs/designations available in the employees table.

```
    :- SELECT DISTINCT job_id FROM employee;
```

10. Write a query to display the name, including first_name, last_name, salary and 15% of salary as PF of all employees.

```
    :-SELECT first_name, last_name, salary, 0.15*salary as pf FROM employee;
```

<-------------------------------- QUERIES BASED ON JOINS
--->

1.Write a query to make a join with employees and departments table to find the name of the employee,
including first_name and last name, department ID and name of departments.

```
    :-  SELECT e.first_name, e.last_name, e.department_id, d.department_name
        FROM employee e
        JOIN departments d ON e.department_id = d.department_id;
```

2.Write a SQL query to make a join with three tables employees, departments and locations to find the name,
 including first_name and last_name, jobs, department name and ID, of the employees working in London.

```
    :-SELECT e.first_name, e.last_name, e.job_id, d.department_id
      FROM employees e
      JOIN departments d ON e.department_id = d.department_id
      JOIN locations l ON d.location_id = l.location_id
      WHERE l.city = 'London';
```

3.Write a query to make a join with a table employees and itself to find the name, including first_name and
 last_name and hire date for those employees who were hired after the employee Jones.

```
    :-  SELECT e.first_name, e.last_name, e.hire_date
        FROM employee e
        JOIN employee j
        ON e.hire_date > j.hire_date AND j.last_name = 'Jones'
```

4.Write a query to make a join with two tables employees and departments to get the department name and number
 of employees working in each department.

```
    :-  SELECT d.department_name, COUNT(e.employee_id) as number_of_employees
        FROM employee e
        JOIN departments d
        ON e.department_id = d.department_id
        GROUP BY d.department_name;
```

5.Write a query to make a join with two tables employees and departments to display the department ID, department
 name and the first name of the manager.

    :-

6. Write a query to make a join with two tables employees and jobs to display the job title and average salary of employees.

```
    :-  SELECT jobs.job_title, AVG(employees.salary) AS avg_salary
        FROM employee
        JOIN jobs
        ON employee.job_id = jobs.job_id
        GROUP BY jobs.job_title;
```

7. Write a query to make a join with two tables job_history and employees to display the status of employees who is currently drawing the salary above 10000.

```
    :-  SELECT employee.status
        FROM employee
        JOIN job_history
        ON employee.employee_id = job_history.employee_id
        WHERE employee.salary > 10000
        AND job_history.end_date IS NULL;
```

<---SUBQUERIES COMMANDS
--->

1. Write a query to find the first_name, last_name and salaries of the employees who have a higher salary than the employee whose last_name is Bull.

```
    :-SELECT first_name, last_name, salary
      FROM employee WHERE salary > (
        SELECT salary FROM employee
        WHERE last_name = 'Bull'
        );
```

2. Write a SQL subquery to find the first_name and last_name of all employees who works in the IT department.

```
    :-SELECT first_name, last_name FROM employees WHERE department = 'IT';
```

3. Write a SQL subquery to find the first_name and last_name of the employees under a manager who works for a department based in the United States.

```
    :-  SELECT first_name, last_name
        FROM employees WHERE manager_id IN (
                    SELECT employee_id FROM employees
                    WHERE department_id IN (
                                    SELECT department_id
                                    FROM departments
                                    WHERE country_id = 'US'
        )
        );
```

4. Write a SQL subquery to find the first_name and last_name of the employees who are working as a manager.

```
    :- SELECT first_name, last_name FROM employees
       WHERE employee_id IN (
                SELECT DISTINCT manager_id
                FROM employees
                WHERE manager_id IS NOT NULL
        );
```

5. Write a SQL subquery to find the first_name, last_name and salary, which is greater than the average salary of the employees.

```
    :-  SELECT first_name, last_name, salary FROM employees
        WHERE salary > (
            SELECT AVG(salary)
            FROM employees
        );
```

6. Write a SQL subquery to find the first_name, last_name and salary, which is equal to the minimum salary for this post, he/she is working on.

```
    :-   SELECT first_name, last_name, salary FROM employees
         WHERE salary = (
                SELECT MIN(salary)
                FROM employees
                WHERE job_id = employees.job_id
         );
```

7. Write a SQL Subquery to find the first_name, last_name and salary of the employees who earn more than the average salary and works in any of the IT departments.

```
    :- SELECT first_name, last_name, salary FROM employees
       WHERE salary >(
                SELECT AVG(salary)
                FROM employees
                WHERE department_id IN (
                        SELECT department_id
                        FROM departments
                        WHERE department_name LIKE '%IT%'
                       )
            );
```

8. Write a SQL subquery to find the first_name, last_name and salary of the employees who draw a more salary than the employee, which the last name is Bell.

```
    :-  SELECT first_name, last_name, salary FROM employee
        WHERE salary > (
                    SELECT salary FROM employees
                    WHERE last_name = 'Bell'
                    LIMIT 1
        );
```

9. Write a SQL subquery to find all the information of the employees who draws the same salary as the minimum salary for all departments.

```
    :-  SELECT * FROM employee
        WHERE salary =(
            SELECT MIN(salary)
            FROM employees
            GROUP BY department
            ORDER BY MIN(salary)
            LIMIT 1
        );
```

10. Write a SQL subquery to find all the information of the employees whose salary greater than the average salary of all departments.

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

11. Write a subquery to find the first_name, last_name, job_id and salary of the employees who draws a salary that is higher than the salary of all the Shipping Clerk (JOB_ID = 'SH_CLERK'). Sort the results on salary from the lowest to highest.

```
    :- SELECT first_name, last_name, job_id, salary FROM employees
        WHERE salary > (
        SELECT MAX(salary)
        FROM employees
        WHERE job_id = 'SH_CLERK'
        )
        ORDER BY salary ASC;
```