Lab sheet 7
PMDS506P Database Management systems.

Q1. Create the following tables and answer the following questions. The customer's table is,

emp_id	emp_name	salary	hire_date	department
1	Alice	60000	2022-01-15	HR
2	Bob	55000	2021-03-22	IT
3	Charlie	70000	2020-07-19	IT
4	Diana	80000	2023-02-10	Finance
5	Eve	65000	2019-11-05	HR
6	Frank	72000	2022-08-30	Finance

1. Write a PL/SQL block that computes the total salary of the employees in the customer table. (use for loop)
2. Write a PL/SQL block that checks if an employee's salary is greater than or equal to 70,000. If it is, print "High salary". Otherwise, print "Regular salary". (use if else)
3. Write a PL/SQL block that uses a FOR loop to display the salaries of employees in the 'IT' department. (for loop)
4. Write a PL/SQL block that increases the salary of all employees in IT department by 5% and prints the updated salaries. (for loop)
5. Categorize employees into 'High', 'Medium', or 'Low' salary brackets based on their salary. (Use case expression)
6. Create a SQL query that uses a CASE expression to display different messages based on the department of employees. Display 'Welcome to HR' for employees in the 'HR' department, 'Tech Team' for employees in the 'IT' department, and 'Finance Team' for employees in the 'Finance' department.

Create the procedures and functions and run the same and see the output.

7. Create a PL/SQL procedure increase_salary that takes an emp_id and a percentage increase and updates the salary of the employee by the given percentage.
8. Write a PL/SQL function get_employee_department that takes an emp_id and returns the department of the employee.
9. Create a procedure update_salary that takes an emp_id and a new salary and updates the employee's salary in the employee's table.

10. Write a function calculate_bonus that takes a salary and returns a bonus amount based on the following criteria:
    10% of salary if the salary is above 70000.
    5% of salary if the salary is between 60000 and 70000.
    1% of salary if below 60000.
11. Create a procedure promote_employee that takes an emp_id and updates the employee's department to 'Management' if their salary is greater than or equal to 70000.
12. Create a function get_employee_details that takes an emp_id and returns the employee's name, salary, and hire date.
13. Write a function compare_salaries that takes two emp_ids and returns the name of the employee with the highest salary.
14. Create a function count_employees_in_department that takes a department name and returns the number of employees in that department.
15. Write a procedure check_hire_date that takes an emp_id and prints a message indicating if the employee was hired in on or after 2022.
16. Create a procedure increase_salary_by_department that takes a department name and a percentage increase and increases the salary of all employees in that department by the given percentage.
17. Write a function highest_salary_in_department that takes a department name and returns the highest salary in that department. (using aggregate functions)
18. Write a function highest_salary_in_department that takes a department name and returns the highest salary in that department. (Without using aggregate functions)