1. **Write an Oracle stored procedure that accepts two numbers as input, adds them together, and returns the sum.**

```
-- Creating the procedure
CREATE OR REPLACE PROCEDURE Add_Two_Numbers (
   num1 IN NUMBER,  -- First input parameter
   num2 IN NUMBER,  -- Second input parameter
   sum OUT NUMBER   -- Output parameter to return the sum
)
IS
BEGIN
   -- Performing the addition and storing the result in the OUT parameter
   sum := num1 + num2;
END;

-----------Executing Procedure----------------
DECLARE
   result NUMBER;  -- Variable to store the output
BEGIN
   -- Call the procedure with input values and capture the output
   Add_Two_Numbers(10, 20, result);
   -- Display the result
   DBMS_OUTPUT.PUT_LINE('The sum is: ' || result);
END;
```

2. **Write an Oracle stored procedure that accepts two numbers as input and returns the maximum of the two numbers.**

```
---------- Creating the procedure

CREATE OR REPLACE PROCEDURE Find_Max (
   num1 IN NUMBER,    -- First input parameter
   num2 IN NUMBER,    -- Second input parameter
   max_num OUT NUMBER  -- Output parameter to return the maximum number
)
IS
BEGIN
   -- Find the maximum of the two numbers
   IF num1 > num2 THEN
      max_num := num1;
   ELSE
      max_num := num2;
   END IF;
END;
```

----------Executing Procedure

```
DECLARE
  result NUMBER;  -- Variable to store the output
BEGIN
  -- Call the procedure with input values and capture the output
  Find_Max(15, 25, result);
  -- Display the result
  DBMS_OUTPUT.PUT_LINE('The maximum number is: ' || result);
END;
```

3. **Write an Oracle stored procedure that updates the salary of an employee based on the employee ID. The new salary is provided as input.**

```
-- Creating the procedure
CREATE OR REPLACE PROCEDURE Update_Salary (
  emp_id IN NUMBER,       -- Employee ID input
  new_salary IN NUMBER    -- New salary input
)
IS
BEGIN
  -- Update the employee salary in the table
  UPDATE employees
  SET salary = new_salary
  WHERE employee_id = emp_id;

  -- Commit the transaction
  COMMIT;
END;
/
```

----Executing Procedure----------

```
BEGIN
  -- Call the procedure to update the salary of employee with ID 101
  Update_Salary(101, 5000);
  DBMS_OUTPUT.PUT_LINE('Salary updated successfully.');
END;
/
```

4. **Write an Oracle stored procedure that deletes an employee record from the employees table based on employee ID.**

```
---------- Creating the procedure
CREATE OR REPLACE PROCEDURE Delete_Employee (
  emp_id IN NUMBER  -- Employee ID input
)
IS
BEGIN
```

```
  -- Delete the employee record
  DELETE FROM employees
  WHERE employee_id = emp_id;

  -- Commit the transaction
  COMMIT;
END;
/

------------Executing Procedure-----------
BEGIN
  -- Call the procedure to delete the employee with ID 102
  Delete_Employee(102);
  DBMS_OUTPUT.PUT_LINE('Employee deleted successfully.');
END;
/
```

5. **Write an Oracle stored procedure that retrieves the first name and salary of an employee based on the employee ID.**

```
-- Creating the procedure
CREATE OR REPLACE PROCEDURE Get_Employee_Details (
  emp_id IN NUMBER,       -- Employee ID input
  emp_name OUT VARCHAR2,   -- Output for employee name
  emp_salary OUT NUMBER    -- Output for employee salary
)
IS
BEGIN
  -- Select employee details
  SELECT first_name, salary
  INTO emp_name, emp_salary
  FROM employees
  WHERE employee_id = emp_id;
END;
/

--------------Execuring Procedure-------

DECLARE
  name VARCHAR2(50);  -- Variable to store the output for name
  salary NUMBER;      -- Variable to store the output for salary
BEGIN
  -- Call the procedure with employee ID 103
  Get_Employee_Details(103, name, salary);
  -- Display the result
  DBMS_OUTPUT.PUT_LINE('Employee Name: ' || name || ', Salary: ' || salary);
END;
/
```

Functions

6. Create a function that accepts two numbers as input and returns their sum.

```
CREATE OR REPLACE FUNCTION add_numbers(
  num1 NUMBER,
  num2 NUMBER
) RETURN NUMBER IS
BEGIN
  RETURN num1 + num2;
END;
/
```

7. **Function to Find Maximum of Two Numbers**

```
CREATE OR REPLACE FUNCTION get_maximum(
  num1 NUMBER,
  num2 NUMBER
) RETURN NUMBER IS
BEGIN
  IF num1 > num2 THEN
    RETURN num1;
  ELSE
    RETURN num2;
  END IF;
END;
/
```

8. Function to Update Employee Salary

```
CREATE OR REPLACE FUNCTION update_employee_salary(
  emp_id NUMBER,
  new_salary NUMBER
) RETURN VARCHAR2 IS
BEGIN
  UPDATE employees
  SET salary = new_salary
  WHERE employee_id = emp_id;

  IF SQL%ROWCOUNT > 0 THEN
    RETURN 'Salary updated successfully';
  ELSE
    RETURN 'Employee not found';
  END IF;
```

```
END;
/
```

9. Function to Delete an Employee Record

```
CREATE OR REPLACE FUNCTION delete_employee(
  emp_id NUMBER
) RETURN VARCHAR2 IS
BEGIN
  DELETE FROM employees
  WHERE employee_id = emp_id;

  IF SQL%ROWCOUNT > 0 THEN
    RETURN 'Employee deleted successfully';
  ELSE
    RETURN 'Employee not found';
  END IF;
END;
/
```

10 . Function to Retrieve Employee's First Name and Salary

```
CREATE OR REPLACE FUNCTION get_employee_details(
  emp_id NUMBER
) RETURN VARCHAR2 IS
  emp_first_name VARCHAR2(100);
  emp_salary NUMBER;
BEGIN
  SELECT first_name, salary
  INTO emp_first_name, emp_salary
  FROM employees
  WHERE employee_id = emp_id;

  RETURN 'First Name: ' || emp_first_name || ', Salary: ' || emp_salary;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RETURN 'Employee not found';
END;
/
```