NAME: KODEESWARAN M

COLLEGE: BANNARI AMMAN INSTITUTE OF TECHNOLOGY

Question 1: Create a Procedure to Insert Employee Data

Write a PL/SQL procedure named insert_employee to insert employee data into the EMPLOYEES table:

- Table structure: EMPLOYEES (EMP_ID NUMBER, EMP_NAME VARCHAR2(100), DEPARTMENT VARCHAR2(50), SALARY NUMBER)

Answer:

```
CREATE OR REPLACE PROCEDURE insert_employee (
   p_emp_id     NUMBER,
   p_emp_name   VARCHAR2,
   p_department VARCHAR2,
   p_salary     NUMBER
) AS
BEGIN
   INSERT INTO EMPLOYEES (EMP_ID, EMP_NAME, DEPARTMENT, SALARY)
   VALUES (p_emp_id, p_emp_name, p_department, p_salary);
END;
/
```

To insert values,

```
BEGIN
   insert_employee(1001, 'Ajay', 'Accounts', 32000);
END;
/
```

Question 2: Create a Procedure to Update Employee Salary

Write a PL/SQL procedure named update_salary to update an employee's salary based on their current salary:

- If the current salary is less than 5000, increase it by 10%.
- If the current salary is between 5000 and 10000, increase it by 7.5%.
- If the current salary is more than 10000, increase it by 5%.

Answer:

```
CREATE OR REPLACE PROCEDURE update_salary (
   p_emp_id  NUMBER
) AS
   v_current_salary EMPLOYEES.SALARY%TYPE;
BEGIN
   SELECT SALARY INTO v_current_salary
   FROM EMPLOYEES
   WHERE EMP_ID = p_emp_id;
   IF v_current_salary < 5000 THEN
      v_current_salary := v_current_salary * 1.10;
   ELSIF v_current_salary BETWEEN 5000 AND 10000 THEN
      v_current_salary := v_current_salary * 1.075;
   ELSE
      v_current_salary := v_current_salary * 1.05;
   END IF;

   UPDATE EMPLOYEES
   SET SALARY = v_current_salary
   WHERE EMP_ID = p_emp_id;
END;
/

BEGIN
   update_salary(1001);
END;
/
```

Question 3: Use a Cursor to Display Employee Names

Write a PL/SQL block using a cursor to fetch and display all employee names from the EMPLOYEES table.

Answer:

```
DECLARE
   CURSOR emp_cursor IS
      SELECT EMP_NAME FROM EMPLOYEES;
   v_emp_name EMPLOYEES.EMP_NAME%TYPE;
BEGIN
   OPEN emp_cursor;
   LOOP
      FETCH emp_cursor INTO v_emp_name;
      EXIT WHEN emp_cursor%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE(v_emp_name);
   END LOOP;
   CLOSE emp_cursor;
END;
/
```

Question 4: Create a View for Employees with High Salary

Write a SQL statement to create a view named high_salary_employees that displays employees earning more than 10000.

Answer:

```
CREATE VIEW high_salary_employees AS
SELECT EMP_ID, EMP_NAME, DEPARTMENT, SALARY
FROM EMPLOYEES
WHERE SALARY > 10000;
```

Question 5: Create a Function to Calculate Bonus

Write a PL/SQL function named calculate_bonus to calculate the bonus based on an employee's salary:

- Employees earning less than 5000 get a bonus of 10% of their salary.
- Employees earning between 5000 and 10000 get a bonus of 7.5% of their salary.
- Employees earning more than 10000 get a bonus of 5% of their salary.
-

Answer:

```
CREATE OR REPLACE FUNCTION calculate_bonus(
  p_salary IN NUMBER
) RETURN NUMBER IS
  v_bonus NUMBER;
BEGIN
  IF p_salary < 5000 THEN
    v_bonus := p_salary * 0.1;
  ELSIF p_salary < 10000 THEN
    v_bonus := p_salary * 0.075;
  ELSE
    v_bonus := p_salary * 0.05;
  END IF;

  RETURN v_bonus;
END;
/
```

To call the function,

```
DECLARE
  emp_salary NUMBER := 7800;
  bonus_amount NUMBER;
BEGIN
  bonus_amount := calculate_bonus(emp_salary);
  DBMS_OUTPUT.PUT_LINE('Employee Salary: ' || emp_salary);
  DBMS_OUTPUT.PUT_LINE('Bonus Amount: ' || bonus_amount);
END;
/
```

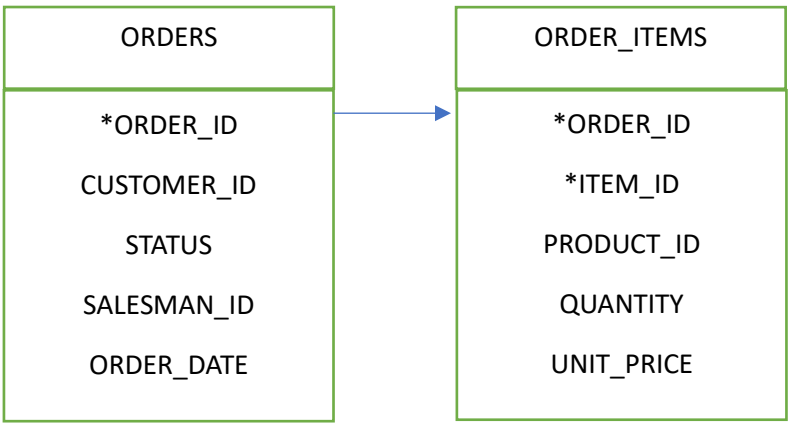
Question 6: Create a Trigger to Log Employee Insertions

Write a PL/SQL trigger named log_employee_insert to log whenever an employee is inserted into the EMPLOYEES table.

Answer:

```
CREATE OR REPLACE TRIGGER log_employee_insert
AFTER INSERT ON EMPLOYEES
FOR EACH ROW
BEGIN
  INSERT INTO EMPLOYEE_LOG (employee_id, insert_timestamp)
    VALUES (:NEW.employee_id, SYSDATE);
END;
/
```

Question 7:Consider the orders and order_items tables from the sample database.



A) Create a view that returns the sales revenues by customers. The values of the credit column are 5% of the total sales revenues.

```
CREATE OR REPLACE VIEW customer_sales_revenue AS
SELECT o.customer_id,
    SUM(oi.quantity * oi.unit_price) AS total_sales,
    SUM(oi.quantity * oi.unit_price) * 0.05 AS credit
FROM orders o
INNER JOIN order_items oi ON o.order_id = oi.order_id
GROUP BY o.customer_id;
```

B) Write the PL/SQL query to develop an anonymous block which:

1. Reset the credit limits of all customers to zero.
2. Fetch customers sorted by sales in descending order and give them new credit limits from a budget of 1 million.

```
DECLARE
  total_budget NUMBER := 1000000;  -- Budget of 1 million
  remaining_budget NUMBER := total_budget;
  customer_id NUMBER;
  sales_amount NUMBER;
  credit_limit NUMBER;
BEGIN
  UPDATE customers
  SET credit_limit = 0;

  FOR rec IN (
    SELECT customer_id, SUM(quantity * unit_price) AS total_sales
    FROM orders o
    INNER JOIN order_items oi ON o.order_id = oi.order_id
    GROUP BY customer_id
    ORDER BY total_sales DESC
  ) LOOP
    customer_id := rec.customer_id;
    sales_amount := rec.total_sales;

    credit_limit := remaining_budget * (sales_amount / total_budget);

    UPDATE customers
    SET credit_limit = credit_limit
    WHERE customer_id = customer_id;
       remaining_budget := remaining_budget - credit_limit;
    EXIT WHEN remaining_budget <= 0;
  END LOOP;

  DBMS_OUTPUT.PUT_LINE('Credit limits updated successfully!');
END; /
```

Question 8: Write a program in PL/SQL to show the uses of implicit cursor without using any attribute.

Answer:

```
DECLARE
  emp_id NUMBER;
BEGIN
  DBMS_OUTPUT.PUT_LINE('Enter employee ID: ');
  emp_id := TO_NUMBER(DBMS_INPUT.GET_LINE);

  SELECT first_name, last_name
  FROM employees
  WHERE employee_id = emp_id;

  IF SQL%FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Employee details:');
    DBMS_OUTPUT.PUT_LINE('  First Name: ' || FIRST_NAME);
    DBMS_OUTPUT.PUT_LINE('  Last Name: ' || LAST_NAME);
  ELSE
    DBMS_OUTPUT.PUT_LINE('Employee with ID ' || emp_id || ' not found.');
  END IF;
END;
/
```

Question 9: Write a program in PL/SQL to create a cursor displays the name and salary of each employee in the EMPLOYEES table whose salary is less than that specified by a passed-in parameter value.

Answer:

```
DECLARE
  CURSOR emp_cursor IS
    SELECT first_name, last_name, salary
    FROM employees
    WHERE salary < :max_salary;  -- Parameter for maximum salary

  emp_record emp_cursor%ROWTYPE;

  max_salary NUMBER;
BEGIN
  DBMS_OUTPUT.PUT_LINE('Enter maximum salary: ');
  max_salary := TO_NUMBER(DBMS_INPUT.GET_LINE);

  OPEN emp_cursor(max_salary);

  LOOP
    FETCH emp_cursor INTO emp_record;
    EXIT WHEN emp_cursor%NOTFOUND;  -- Exit when no more rows

    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || emp_record.first_name || ' ' ||
emp_record.last_name);
    DBMS_OUTPUT.PUT_LINE('Salary: ' || emp_record.salary);
  END LOOP;

  CLOSE emp_cursor;  -- Close the cursor
END;
/
```

Question 10: Write a code in PL/SQL to create a trigger that checks for duplicate values in a specific column and raises an exception if found.

Answer:

```
CREATE OR REPLACE TRIGGER prevent_duplicate_values
BEFORE INSERT ON table_name
FOR EACH ROW
DECLARE
  v_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_count
  FROM table_name
  WHERE column_name = :NEW.column_name;

  IF v_count > 0 THEN
    RAISE_APPLICATION_ERROR(-20001, 'Duplicate value found in ' || column_name);
  END IF;
END;
/
```

Question 11: Write a PL/SQL procedure for selecting some records from the database using some parameters as filters.

- Consider that we are fetching details of employees from ib_employee table where salary is a parameter for filter.

Answer:

```
CREATE OR REPLACE PROCEDURE get_employees_by_salary (
  p_min_salary IN NUMBER,
  p_max_salary IN NUMBER DEFAULT NULL
)
IS
  CURSOR emp_cursor IS
    SELECT employee_id, first_name, last_name, salary
    FROM ib_employee
    WHERE salary >= p_min_salary
      AND (p_max_salary IS NULL OR salary <= p_max_salary);
```

```
  emp_record emp_cursor%ROWTYPE;
BEGIN
  OPEN emp_cursor;
  LOOP
    FETCH emp_cursor INTO emp_record;
    EXIT WHEN emp_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp_record.employee_id);
    DBMS_OUTPUT.PUT_LINE('  Name: ' || emp_record.first_name || ' ' ||
emp_record.last_name);
    DBMS_OUTPUT.PUT_LINE('  Salary: ' || emp_record.salary);
  END LOOP;
  CLOSE emp_cursor;

  IF SQL%NOTFOUND THEN
    DBMS_OUTPUT.PUT_LINE('No employees found with the specified salary range.');
  END IF;
END;
/
```

To get records from Procedure,

```
EXECUTE get_employees_by_salary(25000, 35000);
```

Question 12: Write PL/SQL code block to increment the employee's salary by 1000 whose employee_id is 102 from the given table below.

Answer:

```
BEGIN
  UPDATE employee
  SET salary = salary + 1000
  WHERE employee_id = 102;
END;
/
```