NAME: JAIKANTH S
COLLEGE: BIT

PL/SQL ASSIGNMENT

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)

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

```
CREATE OR REPLACE PROCEDURE update_salary (
    p_emp_id IN NUMBER
) AS
    v_current_salary EMPLOYEES.SALARY%TYPE;
    v_new_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_new_salary := v_current_salary * 1.10;
    ELSIF v_current_salary BETWEEN 5000 AND 10000 THEN
        v_new_salary := v_current_salary * 1.075;
    ELSE
```

```
        v_new_salary := v_current_salary * 1.05;
    END IF;
    UPDATE EMPLOYEES
    SET SALARY = v_new_salary
    WHERE EMP_ID = p_emp_id;
    COMMIT;
  END;
  /
```

Cursors

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.

```
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;
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
    IF emp_cursor%ISOPEN THEN
      CLOSE emp_cursor;
    END IF;
END;
/
```

Views

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.

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

Functions

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.

```
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 * 0.10;
   ELSIF p_salary BETWEEN 5000 AND 10000 THEN
      v_bonus := p_salary * 0.075;
   ELSE
      v_bonus := p_salary * 0.05;
   END IF;
   RETURN v_bonus;
EXCEPTION
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR(-20001, 'An error occurred: ' || SQLERRM);
      RETURN NULL;
END calculate_bonus;
/
```

Triggers

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.

```
CREATE TABLE EMPLOYEE_LOG (
   LOG_ID NUMBER GENERATED BY DEFAULT AS IDENTITY,
   EMP_ID NUMBER,
   EMP_NAME VARCHAR2(100),
   DEPARTMENT VARCHAR2(50),
   SALARY NUMBER,
   INSERTED_AT TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

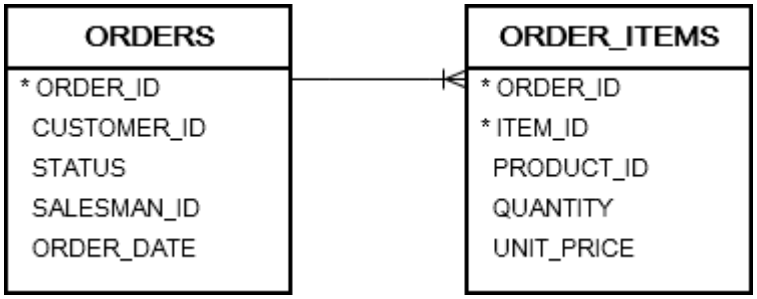
```
   PRIMARY KEY (LOG_ID)
);

CREATE OR REPLACE TRIGGER log_employee_insert
AFTER INSERT ON EMPLOYEES
FOR EACH ROW
BEGIN
   INSERT INTO EMPLOYEE_LOG (EMP_ID, EMP_NAME, DEPARTMENT, SALARY)
   VALUES (:NEW.EMP_ID, :NEW.EMP_NAME, :NEW.DEPARTMENT, :NEW.SALARY);

END;
/

INSERT INTO EMPLOYEES (EMP_ID, EMP_NAME, DEPARTMENT, SALARY)
VALUES (101, 'Jai', 'HR', 50000);
```

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 sales_revenues_by_customers AS
SELECT
 c.customer_id,
 c.customer_name,
 SUM(oi.quantity * oi.unit_price) AS total_sales,
 SUM(oi.quantity * oi.unit_price) * 0.05 AS credit
FROM
 customers c
JOIN
 orders o ON c.customer_id = o.customer_id
JOIN
 order_items oi ON o.order_id = oi.order_id
GROUP BY
 c.customer_id, c.customer_name;
```

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
```

```
v_budget NUMBER := 1000000;
CURSOR cust_cursor IS
SELECT customer_id FROM sales_revenues_by_customers ORDER BY total_sales DESC;
v_customer_id sales_revenues_by_customers.customer_id%TYPE;
BEGIN
-- Reset credit limits
UPDATE customers SET credit_limit = 0;
OPEN cust_cursor;
LOOP
FETCH cust_cursor INTO v_customer_id;
EXIT WHEN cust_cursor%NOTFOUND;
-- Update new credit limit
UPDATE customers
SET credit_limit = credit_limit + (v_budget / (SELECT COUNT(*) FROM
sales_revenues_by_customers))
WHERE customer_id = v_customer_id;
v_budget := v_budget - (v_budget / (SELECT COUNT(*) FROM sales_revenues_by_customers));
END LOOP;
CLOSE cust_cursor;
END;
/
```

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

Table: employees

employee_id	integer
first_name	varchar(25)
last_name	varchar(25)
email	archar(25)
phone_number	varchar(15)
hire_date	date
job_id	varchar(25)
salary	integer
commission_pct	decimal(5,2)
manager_id	integer
department_id	integer

```
DECLARE
v_count INTEGER;
BEGIN
SELECT COUNT(*) INTO v_count FROM employees;
DBMS_OUTPUT.PUT_LINE('Total number of employees: ' || v_count);
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.

Table: employees

employee_id	integer
first_name	varchar(25)
last_name	varchar(25)
email	archar(25)

phone_number	varchar(15)
hire_date	date
job_id	varchar(25)
salary	integer
commission_pct	decimal(5,2)
manager_id	integer
department_id	integer

```
DECLARE
 CURSOR emp_cursor (p_salary NUMBER) IS
 SELECT first_name, last_name, salary
 FROM employees
 WHERE salary < p_salary;
 v_first_name employees.first_name%TYPE;
 v_last_name employees.last_name%TYPE;
 v_salary employees.salary%TYPE;
BEGIN
 OPEN emp_cursor(10000);
 LOOP
 FETCH emp_cursor INTO v_first_name, v_last_name, v_salary;
 EXIT WHEN emp_cursor%NOTFOUND;
 DBMS_OUTPUT.PUT_LINE(v_first_name || ' ' || v_last_name || ': ' || v_salary);
 END LOOP;
 CLOSE emp_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.

```
CREATE OR REPLACE TRIGGER check_duplicate_emp_id
BEFORE INSERT OR UPDATE ON employees
FOR EACH ROW
DECLARE
 v_count INTEGER;
BEGIN
 SELECT COUNT(*)
 INTO v_count
 FROM employees
 WHERE employee_id = :NEW.employee_id;
```

```
 IF v_count > 0 THEN
 RAISE_APPLICATION_ERROR(-20001, 'Duplicate employee_id found.');
 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.

```
CREATE OR REPLACE PROCEDURE select_employees_by_salary (
 p_salary NUMBER
) AS
BEGIN
 FOR emp IN (SELECT * FROM ib_employee WHERE salary = p_salary) LOOP
 DBMS_OUTPUT.PUT_LINE(emp.first_name || ' ' || emp.last_name || ': ' || emp.salary);
 END LOOP;
END;
/
```

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.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL_ID	PHONE_NUMBER	JOIN_DATE	JOB_ID	SALARY
100	ABC	DEF	abef	9876543210	2020-06-06	AD_PRES	24000.00
101	GHI	JKL	ghkl	9876543211	2021-02-08	AD_VP	17000.00
102	MNO	PQR	mnqr	9876543212	2016-05-14	AD_VP	17000.00
103	STU	VWX	stwx	9876543213	2019-06-24	IT_PROG	9000.00

```
BEGIN
 UPDATE EMPLOYEES
 SET SALARY = SALARY + 1000
 WHERE EMPLOYEE_ID = 102;
END;
```