

**Title :**PL/SQL ASSIGNMENT

**Author :** KAMAL B

**Created At :** 17-07-24

**Last Modified Date :** 23-04-24

**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 table EMPLOYEES (  
EMP_ID NUMBER,  
EMP_NAME VARCHAR2(100),  
DEPARTMENT VARCHAR2(50),  
SALARY NUMBER  
);  
SET SERVEROUTPUT ON;  
DESC EMPLOYEES;
```

```
CREATE OR REPLACE PROCEDURE insert_employee  
AS  
BEGIN  
INSERT INTO EMPLOYEES VALUES(1,'Arul','CSE',38000);  
INSERT INTO EMPLOYEES VALUES(2,'sanjay','CSE',3000);  
INSERT INTO EMPLOYEES VALUES(3,'Gayathri','CSE',7000);  
INSERT INTO EMPLOYEES VALUES(4,'Saran','CSE',8000);  
INSERT INTO EMPLOYEES VALUES(5,'Anbu','CSE',10000);  
COMMIT;  
END;  
  
EXECUTE insert_employee;  
SELECT * FROM EMPLOYEES;  
TRUNCATE TABLE EMPLOYEES;
```

**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(up_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 = up_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=up_emp_id;
COMMIT;
END;

EXECUTE update_salary(1);
SELECT * FROM EMPLOYEES;
```

**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
v_emp_name EMPLOYEES.EMP_NAME%TYPE;
```

```

CURSOR C1
IS
SELECT EMP_NAME FROM EMPLOYEES;
BEGIN
OPEN C1;
LOOP
FETCH C1 INTO v_emp_name;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(v_emp_name);
end loop;
Close c1;
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.**

```

CREATE OR REPLACE VIEW high_salary_employees
AS
SELECT SALARY FROM EMPLOYEES WHERE SALARY > 10000;

SELECT * FROM high_salary_employees;

```

**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(f_salary IN NUMBER)
RETURN NUMBER
AS
v_cal_bonus NUMBER;
BEGIN
IF f_salary < 5000 THEN

```

```

        v_cal_bonus := f_salary * 0.10;
ELSIF f_salary BETWEEN 5000 AND 10000 THEN
        v_cal_bonus := f_salary * 0.075;
ELSE
        v_cal_bonus :=f_salary * 0.05;
END IF;
RETURN v_cal_bonus;
END calculate_bonus;

SELECT calculate_bonus(13000) FROM EMPLOYEES;

```

**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 OR REPLACE TRIGGER log_employee_insert
BEFORE INSERT ON EMPLOYEES
FOR EACH ROW
ENABLE
DECLARE
v_emp_insert VARCHAR2(20);
BEGIN
SELECT EMP_NAME INTO v_emp_insert FROM EMPLOYEES;
DBMS_OUTPUT.PUT_LINE(v_emp_insert);
END;

INSERT INTO EMPLOYEES VALUES(6,'SRIYAZHINI','CSE',13000);

```

**7. consider the order and order\_items tables from the sample database.**

```

CREATE TABLE ORDERS (
    ORDER_ID NUMBER PRIMARY KEY,

```

```
CUSTOMER_ID NUMBER,  
STATUS VARCHAR2(50),  
SALESMAN_ID NUMBER,  
ORDER_DATE DATE  
);
```

```
DROP TABLE ORDERS;
```

```
CREATE TABLE ORDER_ITEMS (  
    ORDER_ID NUMBER PRIMARY KEY,  
    ITEM_ID NUMBER ,  
    PRODUCT_ID NUMBER,  
    QUANTITY NUMBER,  
    UNIT_PRICE NUMBER,  
    FOREIGN KEY (ORDER_ID) REFERENCES ORDERS(ORDER_ID)  
);
```

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

```
CREATE TABLE EMPLOYEES_DETAILS (  
    EMPLOYEE_ID NUMBER,  
    FIRST_NAME VARCHAR2(50),  
    LAST_NAME VARCHAR2(50),  
    EMAIL VARCHAR2(100),  
    PHONE_NUMBER VARCHAR2(20),  
    HIRE_DATE DATE,  
    JOB_ID VARCHAR2(10),  
    SALARY NUMBER,  
    COMMISSION_PCT NUMBER,  
    MANAGER_ID NUMBER,  
    DEPARTMENT_ID NUMBER  
);
```

```
INSERT INTO EMPLOYEES_DETAILS (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID,  
DEPARTMENT_ID)
```

```
VALUES (1, 'Arul', 'Praba', 'arul06@gmail.com', '9876543210', TO_DATE('2002-09-20', 'YYYY-MM-DD'), 'IT_PROG', 65000, 0.15, 2, 1);
```

```
INSERT INTO EMPLOYEES_DETAILS (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID,  
DEPARTMENT_ID)
```

```
VALUES (2, 'Sanjay', 'Kumar', 'sanjay@gmail.com', '9876543219', TO_DATE('2003-03-27', 'YYYY-MM-DD'), 'HR_REP', 40000, 0.08, 1, 2);
```

```
INSERT INTO EMPLOYEES_DETAILS (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID,  
DEPARTMENT_ID)
```

```
VALUES (3, 'Gayathri', 'Iyer', 'gayathri@gmail.com', '9876543218', TO_DATE('2002-11-14', 'YYYY-MM-DD'), 'AD_VP', 95000, 0.12, 2, 3);
```

```
INSERT INTO EMPLOYEES_DETAILS (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID,  
DEPARTMENT_ID)
```

```
VALUES (4, 'Saran', 'Ravi', 'saran@gmail.com', '9876543217', TO_DATE('2004-03-21', 'YYYY-MM-DD'), 'IT_MGR', 78000, 0.10, 3, 4);
```

```
INSERT INTO EMPLOYEES_DETAILS (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID,  
DEPARTMENT_ID)
```

```
VALUES (5, 'Anbu', 'Kumar', 'anbu@gmail.com', '9876543216', TO_DATE('2005-07-10', 'YYYY-MM-DD'), 'FIN_MGR', 82000, 0.09, 4, 5);
```

```
COMMIT;
```

```
DROP TABLE EMPLOYEES_DETAILS;
```

```
SELECT * FROM EMPLOYEES_DETAILS;
```

```

DECLARE

CURSOR C2

IS

SELECT EMPLOYEE_ID,FIRST_NAME,LAST_NAME FROM EMPLOYEES_DETAILS;

v_employee_id EMPLOYEES_DETAILS.EMPLOYEE_ID%TYPE;

v_first_name EMPLOYEES_DETAILS.FIRST_NAME%TYPE;

v_last_name EMPLOYEES_DETAILS.LAST_NAME%TYPE;

BEGIN

OPEN C2;

LOOP

FETCH C2 INTO v_employee_id,v_first_name,v_last_name;

EXIT WHEN C2%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('Employee_ID: ' || v_employee_id);

DBMS_OUTPUT.PUT_LINE('First_Name: ' || v_first_name);

DBMS_OUTPUT.PUT_LINE('Last_Name: ' || v_last_name);

END LOOP;

CLOSE C2;

END;

/

```

**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.**

```

DECLARE

CURSOR C3

IS

SELECT FIRST_NAME,LAST_NAME,SALARY FROM EMPLOYEES_DETAILS;

v_first_name EMPLOYEES_DETAILS.FIRST_NAME%TYPE;

v_last_name EMPLOYEES_DETAILS.LAST_NAME%TYPE;

```

```

v_salary EMPLOYEES_DETAILS.SALARY%TYPE;

BEGIN

OPEN C3;

LOOP

FETCH C3 INTO v_first_name,v_last_name,v_salary;

EXIT WHEN C3%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('First_Name: ' || v_first_name);

DBMS_OUTPUT.PUT_LINE('Last_Name: ' || v_last_name);

DBMS_OUTPUT.PUT_LINE('Salary: ' || v_salary);

end loop;

Close c3;

end;

```

**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 trg_check_duplicate_email

BEFORE INSERT OR UPDATE ON EMPLOYEES_DETAILS

FOR EACH ROW

DECLARE

    v_count NUMBER;

BEGIN

SELECT COUNT(*) INTO v_count FROM EMPLOYEES_DETAILS WHERE EMAIL = :NEW.EMAIL AND
EMPLOYEE_ID <> :NEW.EMPLOYEE_ID;

IF v_count > 0 THEN

    DBMS_OUTPUT.PUT_LINE('Duplicate Occurs');

ELSE

    DBMS_OUTPUT.PUT_LINE('No Duplicate Occurs');

END IF;

END;

/

```



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

```
CREATE OR REPLACE PROCEDURE get_employees_by_salary(p_salary IN NUMBER)
IS
CURSOR emp_cursor IS
SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID,
SALARY, COMMISSION_PCT, MANAGER_ID, DEPARTMENT_ID
FROM EMPLOYEES_DETAILS
WHERE SALARY > p_salary;

v_employee_id EMPLOYEES_DETAILS.EMPLOYEE_ID%TYPE;
v_first_name EMPLOYEES_DETAILS.FIRST_NAME%TYPE;
v_last_name EMPLOYEES_DETAILS.LAST_NAME%TYPE;
v_email EMPLOYEES_DETAILS.EMAIL%TYPE;
v_phone_number EMPLOYEES_DETAILS.PHONE_NUMBER%TYPE;
v_hire_date EMPLOYEES_DETAILS.HIRE_DATE%TYPE;
v_job_id EMPLOYEES_DETAILS.JOB_ID%TYPE;
v_salary EMPLOYEES_DETAILS.SALARY%TYPE;
v_commission_pct EMPLOYEES_DETAILS.COMMISSION_PCT%TYPE;
v_manager_id EMPLOYEES_DETAILS.MANAGER_ID%TYPE;
v_department_id EMPLOYEES_DETAILS.DEPARTMENT_ID%TYPE;
BEGIN
OPEN emp_cursor;
LOOP
    FETCH emp_cursor INTO v_employee_id, v_first_name, v_last_name, v_email, v_phone_number,
v_hire_date, v_job_id, v_salary, v_commission_pct, v_manager_id, v_department_id;

    EXIT WHEN emp_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_employee_id);
    DBMS_OUTPUT.PUT_LINE('First Name: ' || v_first_name);
```

```

DBMS_OUTPUT.PUT_LINE('Last Name: ' || v_last_name);
DBMS_OUTPUT.PUT_LINE('Email: ' || v_email);
DBMS_OUTPUT.PUT_LINE('Phone Number: ' || v_phone_number);
DBMS_OUTPUT.PUT_LINE('Hire Date: ' || TO_CHAR(v_hire_date, 'YYYY-MM-DD'));
DBMS_OUTPUT.PUT_LINE('Job ID: ' || v_job_id);
DBMS_OUTPUT.PUT_LINE('Salary: ' || v_salary);
DBMS_OUTPUT.PUT_LINE('Commission Percentage: ' || v_commission_pct);
DBMS_OUTPUT.PUT_LINE('Manager ID: ' || v_manager_id);
DBMS_OUTPUT.PUT_LINE('Department ID: ' || v_department_id);

END LOOP;

CLOSE emp_cursor;

END get_employees_by_salary;

/

EXECUTE get_employees_by_salary(55000);

```

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

```

CREATE TABLE EMPLOYEES_DATA(
    EMPLOYEE_ID NUMBER,
    FIRST_NAME VARCHAR2(50),
    LAST_NAME VARCHAR2(50),
    EMAIL VARCHAR2(100),
    PHONE_NUMBER VARCHAR2(20),
    JOIN_DATE DATE,
    JOB_ID VARCHAR2(10),
    SALARY NUMBER
);

INSERT INTO EMPLOYEES_DATA (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,
PHONE_NUMBER, JOIN_DATE, JOB_ID, SALARY)

```

```
VALUES (100, 'Arul', 'Praba', 'arul.praba@gmail.com', '9876543210', TO_DATE('2020-06-06', 'YYYY-MM-DD'), 'IT_PROG', 26000.00);
```

```
INSERT INTO EMPLOYEES_DATA (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, JOIN_DATE, JOB_ID, SALARY)
```

```
VALUES (101, 'Sanjay', 'Kumar', 'sanjay.kumar@gmail.com', '9876543211', TO_DATE('2021-02-08',  
'YYYY-MM-DD'), 'HR_REP', 18000.00);
```

```
INSERT INTO EMPLOYEES_DATA (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, JOIN_DATE, JOB_ID, SALARY)
```

```
VALUES (102, 'Gayathri', 'Iyer', 'gayathri.iyer@gmail.com', '9876543212', TO_DATE('2016-05-14',  
'YYYY-MM-DD'), 'AD_VP', 19000.00);
```

```
INSERT INTO EMPLOYEES_DATA (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, JOIN_DATE, JOB_ID, SALARY)
```

```
VALUES (103, 'Saran', 'Ravi', 'saran.ravi@gmail.com', '9876543213', TO_DATE('2019-06-24', 'YYYY-MM-DD'), 'IT_PROG', 10000.00);
```

```
INSERT INTO EMPLOYEES_DATA (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,  
PHONE_NUMBER, JOIN_DATE, JOB_ID, SALARY)
```

```
VALUES (104, 'Anbu', 'Kumar', 'anbu.kumar@gmail.com', '9876543214', TO_DATE('2018-08-30',  
'YYYY-MM-DD'), 'FIN_MGR', 22000.00);
```

```
SELECT * FROM EMPLOYEES_DATA;
```

```
DECLARE
```

```
BEGIN
```

```
UPDATE EMPLOYEES_DATA SET SALARY=SALARY+1000 WHERE EMPLOYEE_ID=102;
```

```
COMMIT;
```

```
END;
```