1. PL/SQL Block to Calculate Grade of Minimum 10 Students

sql

Copy

Edit

BEGIN

FOR i IN 1..10 LOOP

DECLARE

marks NUMBER := DBMS\_RANDOM.VALUE(0, 100);

grade CHAR(1);

BEGIN

IF marks >= 90 THEN

grade := 'A';

ELSIF marks >= 75 THEN

grade := 'B';

ELSIF marks >= 60 THEN

grade := 'C';

ELSIF marks >= 40 THEN

grade := 'D';

ELSE

grade := 'F';

END IF;

DBMS\_OUTPUT.PUT\_LINE('Student ' || i || ': Marks=' || ROUND(marks) || ', Grade=' || grade);

END;

END LOOP;

END;

/

--------------------------------------------------------

2. Procedure to Add Employee and Return Count in Department

sql

Copy

Edit

CREATE OR REPLACE PROCEDURE AddEmployee(

p\_empno NUMBER,

p\_ename VARCHAR2,

p\_sal NUMBER,

p\_deptno NUMBER,

emp\_count OUT NUMBER

)

IS

BEGIN

INSERT INTO emp(empno, ename, sal, deptno)

VALUES (p\_empno, p\_ename, p\_sal, p\_deptno);

SELECT COUNT(\*) INTO emp\_count

FROM emp

WHERE deptno = p\_deptno;

END;

/

--------------------------------------------------------

3. Function to Return Salary Status

sql

Copy

Edit

CREATE OR REPLACE FUNCTION GetSalaryStatus(p\_empno NUMBER)

RETURN VARCHAR2

IS

v\_sal NUMBER;

BEGIN

SELECT sal INTO v\_sal FROM emp WHERE empno = p\_empno;

IF v\_sal < 30000 THEN

RETURN 'Low';

ELSIF v\_sal BETWEEN 30000 AND 60000 THEN

RETURN 'Medium';

ELSE

RETURN 'High';

END IF;

END;

/

--------------------------------------------------------

4. Function to Return Number of Employees in Department

sql

Copy

Edit

CREATE OR REPLACE FUNCTION GetEmpCount(p\_deptno NUMBER)

RETURN NUMBER

IS

v\_count NUMBER;

BEGIN

SELECT COUNT(\*) INTO v\_count

FROM emp

WHERE deptno = p\_deptno;

RETURN v\_count;

END;

/

--------------------------------------------------------

5. Function for Customer Level

sql

Copy

Edit

CREATE OR REPLACE FUNCTION CustomerLevel(p\_total\_purchase NUMBER)

RETURN VARCHAR2

IS

BEGIN

IF p\_total\_purchase > 100000 THEN

RETURN 'Platinum';

ELSIF p\_total\_purchase >= 50000 THEN

RETURN 'Gold';

ELSE

RETURN 'Silver';

END IF;

END;

/

--------------------------------------------------------

6. Function for Even/Odd and Largest of Three

(a) Even or Odd

sql

Copy

Edit

CREATE OR REPLACE FUNCTION IsEven(p\_number NUMBER)

RETURN VARCHAR2

IS

BEGIN

IF MOD(p\_number, 2) = 0 THEN

RETURN 'Even';

ELSE

RETURN 'Odd';

END IF;

END;

/

--------------------------------------------------------

--------------------------------------------------------

(b) Largest of Three Numbers

sql

Copy

Edit

CREATE OR REPLACE FUNCTION LargestOfThree(a NUMBER, b NUMBER, c NUMBER)

RETURN NUMBER

IS

BEGIN

RETURN GREATEST(a, b, c);

END;

/

--------------------------------------------------------

7. Cursor Without and With Handler

Without Handler

sql

Copy

Edit

DECLARE

CURSOR c\_emp IS SELECT ename FROM emp;

v\_name emp.ename%TYPE;

BEGIN

OPEN c\_emp;

LOOP

FETCH c\_emp INTO v\_name;

EXIT WHEN c\_emp%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_name);

END LOOP;

CLOSE c\_emp;

END;

/

With Exception Handler

sql

Copy

Edit

DECLARE

CURSOR c\_emp IS SELECT ename FROM emp;

v\_name emp.ename%TYPE;

BEGIN

OPEN c\_emp;

LOOP

FETCH c\_emp INTO v\_name;

EXIT WHEN c\_emp%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(v\_name);

END LOOP;

CLOSE c\_emp;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred');

END;

/

--------------------------------------------------------

8. Procedure for Arithmetic Operations

sql

Copy

Edit

CREATE OR REPLACE PROCEDURE DoArithmetic(a NUMBER, b NUMBER)

IS

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Addition: ' || (a + b));

DBMS\_OUTPUT.PUT\_LINE('Subtraction: ' || (a - b));

DBMS\_OUTPUT.PUT\_LINE('Multiplication: ' || (a \* b));

DBMS\_OUTPUT.PUT\_LINE('Division: ' || (a / b));

END;

/

--------------------------------------------------------

9. Trigger Before Insert / After Delete

Before Insert

sql

Copy

Edit

CREATE OR REPLACE TRIGGER BeforeInsertEmp

BEFORE INSERT ON emp

FOR EACH ROW

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Inserting new employee: ' || :NEW.ename);

END;

/

After Delete

sql

Copy

Edit

CREATE OR REPLACE TRIGGER AfterDeleteEmp

AFTER DELETE ON emp

FOR EACH ROW

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Deleted employee: ' || :OLD.ename);

END;

/