PL/SQL Practical: Named Block - Stored Procedure and Function (Student Grade Categorization)

Objective:
To create a named PL/SQL stored **function** and **procedure** that categorize students into grades based on their marks. The function returns the grade, and the procedure stores the results into a table with proper exception handling.

```
-- File: grade_procedure_function.sql
-- Purpose: Simple Named PL/SQL solution to classify students into grade categories
-- Run in Oracle APEX SQL Workshop -> SQL Commands (Run Script)

-- 0) Drop tables if exist (safe for dev/test)
BEGIN
  BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE Result CASCADE CONSTRAINTS';
  EXCEPTION WHEN OTHERS THEN
    IF SQLCODE != -942 THEN
      RAISE;
    END IF;
  END;

  BEGIN
    EXECUTE IMMEDIATE 'DROP TABLE Stud_Marks CASCADE CONSTRAINTS';
  EXCEPTION WHEN OTHERS THEN
    IF SQLCODE != -942 THEN
      RAISE;
    END IF;
  END;
END;
/

-- 1) Create input table (Stud_Marks) and output table (Result)
CREATE TABLE Stud_Marks (
  Roll_no     NUMBER PRIMARY KEY,
  Name        VARCHAR2(100),
  Total_Marks NUMBER
);
/

CREATE TABLE Result (
  Roll_no     NUMBER PRIMARY KEY,
  Name        VARCHAR2(100),
  Class       VARCHAR2(50)
);
/

-- 2) Insert sample data
INSERT ALL
  INTO Stud_Marks (Roll_no, Name, Total_Marks) VALUES (1, 'Amit Sharma', 1450)
  INTO Stud_Marks (Roll_no, Name, Total_Marks) VALUES (2, 'Rina Patel', 980)
  INTO Stud_Marks (Roll_no, Name, Total_Marks) VALUES (3, 'Vikram Rao', 860)
  INTO Stud_Marks (Roll_no, Name, Total_Marks) VALUES (4, 'Sanya Mehta', 700)
SELECT * FROM dual;
/

COMMIT;
/

-- 3) Function: fn_get_grade
CREATE OR REPLACE FUNCTION fn_get_grade(p_marks IN NUMBER) RETURN VARCHAR2 IS
  v_class VARCHAR2(50);
  invalid_marks EXCEPTION;
BEGIN
  IF p_marks IS NULL THEN
    RAISE invalid_marks;
  END IF;

  IF p_marks < 0 OR p_marks > 1500 THEN
    RAISE invalid_marks;
  END IF;
  IF p_marks >= 990 AND p_marks <= 1500 THEN
```

```
    v_class := 'Distinction';
  ELSIF p_marks >= 900 AND p_marks <= 989 THEN
    v_class := 'First Class';
  ELSIF p_marks >= 825 AND p_marks <= 899 THEN
    v_class := 'Higher Second Class';
  ELSIF p_marks >= 600 AND p_marks <= 824 THEN
    v_class := 'Second Class';
  ELSE
    v_class := 'Fail';
  END IF;

  RETURN v_class;

EXCEPTION
  WHEN invalid_marks THEN
    RETURN 'Invalid Marks';
  WHEN OTHERS THEN
    RETURN 'Error';
END fn_get_grade;
/

-- 4) Procedure: proc_Grade
CREATE OR REPLACE PROCEDURE proc_Grade(p_roll_no IN NUMBER) IS
  v_name       Stud_Marks.Name%TYPE;
  v_marks      Stud_Marks.Total_Marks%TYPE;
  v_class      VARCHAR2(50);
BEGIN
  SELECT Name, Total_Marks INTO v_name, v_marks
    FROM Stud_Marks
   WHERE Roll_no = p_roll_no;

  v_class := fn_get_grade(v_marks);

  IF v_class = 'Invalid Marks' THEN
    RAISE_APPLICATION_ERROR(-20020, 'Marks are out of valid range for roll ' || p_roll_no);
  END IF;

  BEGIN
    UPDATE Result
       SET Name = v_name, Class = v_class
     WHERE Roll_no = p_roll_no;

    IF SQL%ROWCOUNT = 0 THEN
      INSERT INTO Result (Roll_no, Name, Class)
      VALUES (p_roll_no, v_name, v_class);
    END IF;
  EXCEPTION
    WHEN OTHERS THEN
      ROLLBACK;
      RAISE;
  END;

  COMMIT;

  DBMS_OUTPUT.PUT_LINE('Processed Roll=' || p_roll_no || ', Name=' || v_name || ', Class=' || v_class

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No student found with Roll No: ' || p_roll_no);
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error processing Roll ' || p_roll_no || ': ' || SQLERRM);
    ROLLBACK;
END proc_Grade;
/

-- 5) Example PL/SQL block to execute procedure in Oracle APEX
BEGIN
  proc_Grade(1);
  proc_Grade(2);
END;
/

-- Check Result table
SELECT * FROM Result ORDER BY Roll_no;
/

-- 6) Process all students using a cursor loop
```

```
BEGIN
  FOR r IN (SELECT Roll_no FROM Stud_Marks) LOOP
    BEGIN
      proc_Grade(r.Roll_no);
    EXCEPTION
      WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Failed for Roll=' || r.Roll_no || ': ' || SQLERRM);
    END;
  END LOOP;
END;
/

-- View final results
SELECT * FROM Result ORDER BY Roll_no;
```

Explanation:

1. The **Stud_Marks** table holds student names and total marks.
2. The **Result** table holds the output with grade/class.
3. **fn_get_grade** returns a grade string based on marks ranges.
4. **proc_Grade** uses the function to determine class and update/insert Result.
5. The script includes exception handling for invalid marks and missing records.
6. DBMS_OUTPUT shows messages when run in Oracle APEX (enable “DBMS Output: ON”).