**Program Code –**

ACCEPT v\_roll NUMBER PROMPT 'Enter Roll No: '

ACCEPT v\_book CHAR PROMPT 'Enter Book Name: '

SET SERVEROUTPUT ON;

DECLARE

v\_rollno NUMBER := &v\_roll;

v\_bookname VARCHAR2(50) := '&v\_book';

v\_dateissue DATE;

v\_days NUMBER;

v\_fine NUMBER := 0;

BEGIN

-- Fetch Date of Issue

SELECT DateofIssue INTO v\_dateissue

FROM Borrower

WHERE Roll\_no = v\_rollno AND NameofBook = v\_bookname AND Status = 'I';

-- Days kept

v\_days := TRUNC(SYSDATE - v\_dateissue);

-- Fine Calculation

IF v\_days <= 15 THEN

v\_fine := 0;

ELSIF v\_days > 15 AND v\_days <= 30 THEN

v\_fine := (v\_days - 15) \* 5;

ELSE

v\_fine := (15 \* 5) + ((v\_days - 30) \* 50);

END IF;

-- Update status

UPDATE Borrower

SET Status = 'R'

WHERE Roll\_no = v\_rollno AND NameofBook = v\_bookname;

-- Insert fine if any

IF v\_fine > 0 THEN

INSERT INTO Fine (Roll\_no, FineDate, Amt)

VALUES (v\_rollno, SYSDATE, v\_fine);

END IF;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Book Returned Successfully.');

DBMS\_OUTPUT.PUT\_LINE('Days Kept: ' || v\_days);

DBMS\_OUTPUT.PUT\_LINE('Fine Amount: ' || v\_fine);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No record found for given Roll\_no and Book.');

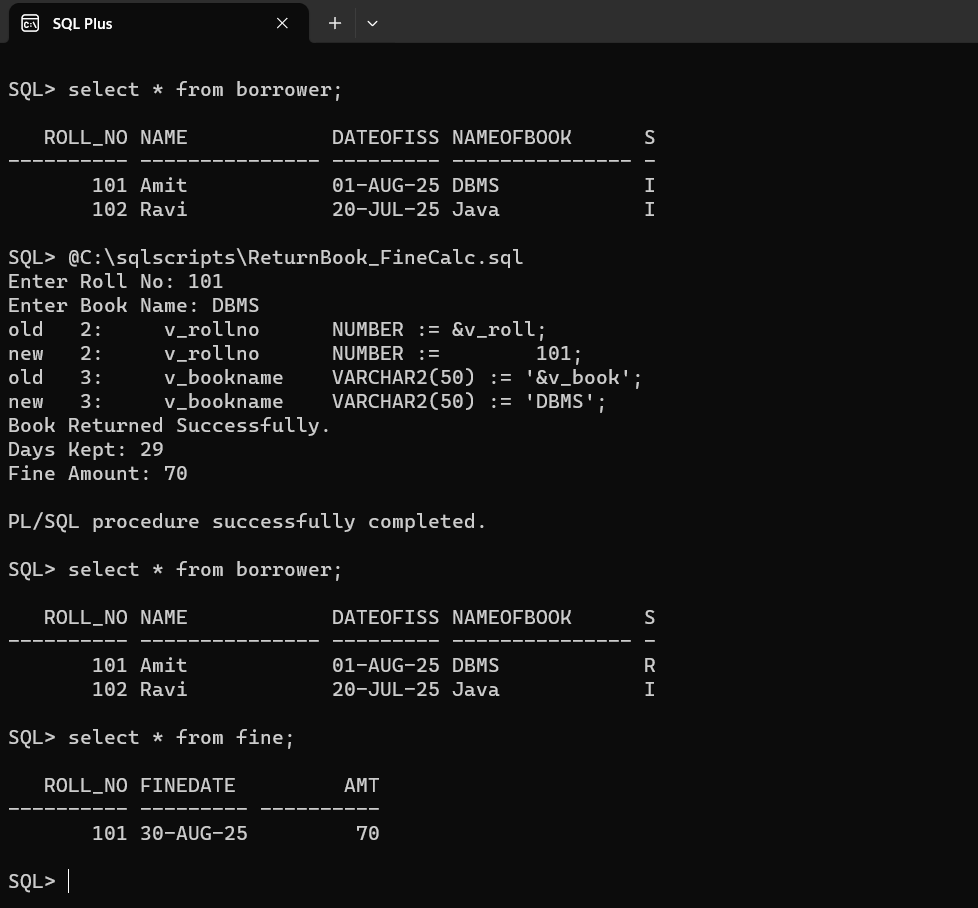
WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END;

/

**Output –**



**Program Code -**

ACCEPT user\_option NUMBER PROMPT 'Enter 1 for FOR loop or 2 for WHILE loop: '

SET SERVEROUTPUT ON;

DECLARE

v\_option NUMBER := &user\_option;

v\_radius NUMBER;

v\_area NUMBER;

BEGIN

CASE v\_option

WHEN 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Using FOR loop...');

FOR r IN 5..9 LOOP

v\_area := 3.14159 \* r \* r;

INSERT INTO areas VALUES (r, v\_area);

END LOOP;

WHEN 2 THEN

DBMS\_OUTPUT.PUT\_LINE('Using WHILE loop...');

v\_radius := 5;

WHILE v\_radius <= 9 LOOP

v\_area := 3.14159 \* v\_radius \* v\_radius;

INSERT INTO areas VALUES (v\_radius, v\_area);

v\_radius := v\_radius + 1;

END LOOP;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Invalid option! Enter 1 or 2.');

END CASE;

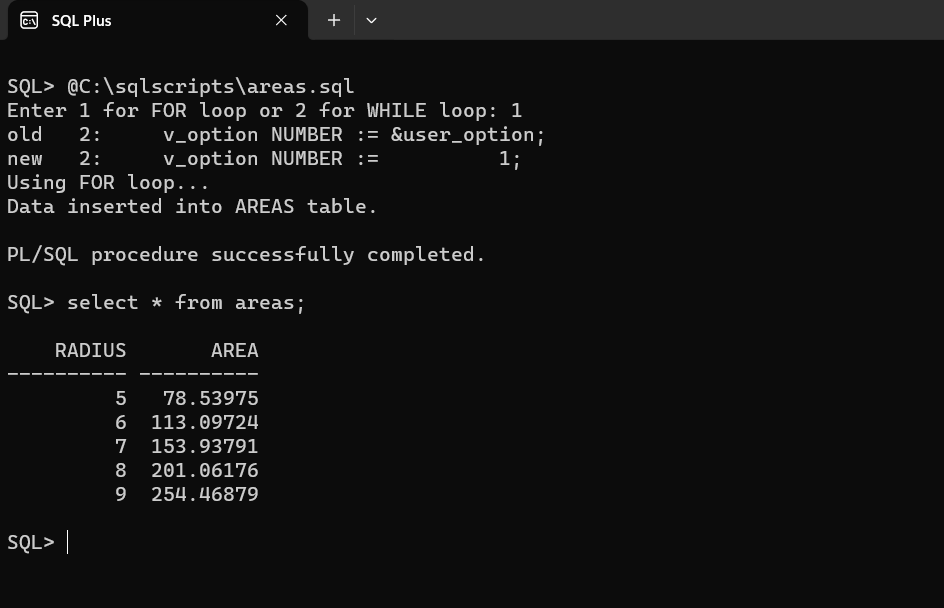
COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Data inserted into AREAS table.');

END;

/

**Output -**



**Program Code -**

SET SERVEROUTPUT ON;

CREATE OR REPLACE FUNCTION fn\_GetGrade(p\_marks NUMBER)

RETURN VARCHAR2 IS

v\_class VARCHAR2(30);

BEGIN

IF p\_marks BETWEEN 990 AND 1500 THEN

v\_class := 'Distinction';

ELSIF p\_marks BETWEEN 900 AND 989 THEN

v\_class := 'First Class';

ELSIF p\_marks BETWEEN 825 AND 899 THEN

v\_class := 'Higher Second Class';

ELSE

v\_class := 'Not Categorized';

END IF;

RETURN v\_class;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'No Data Found';

WHEN OTHERS THEN

RETURN 'Error Occurred';

END;

/

CREATE OR REPLACE PROCEDURE proc\_Grade IS

BEGIN

DELETE FROM Result;

INSERT INTO Result (roll, name, class)

SELECT roll,

name,

fn\_GetGrade(total\_marks)

FROM Stud\_Marks;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No Data Found in Stud\_Marks');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END;

/

BEGIN

proc\_Grade;

DBMS\_OUTPUT.PUT\_LINE('Student Grades Inserted into Result table');

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error in main block: ' || SQLERRM);

END;

/

**Output –**

