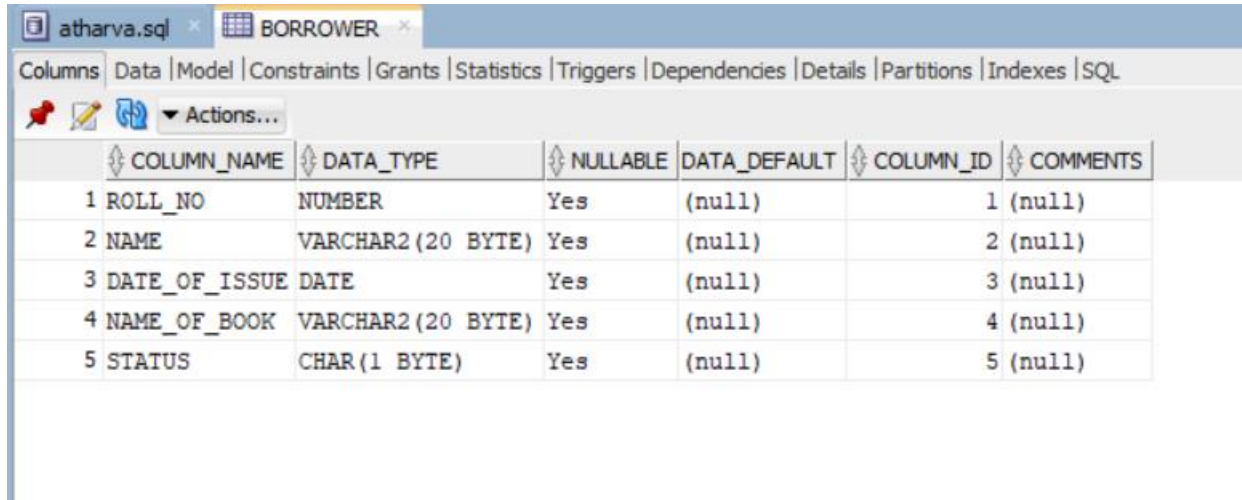


DBMS LAB CHIT SOLUTIONS

(1) Chit 2 –

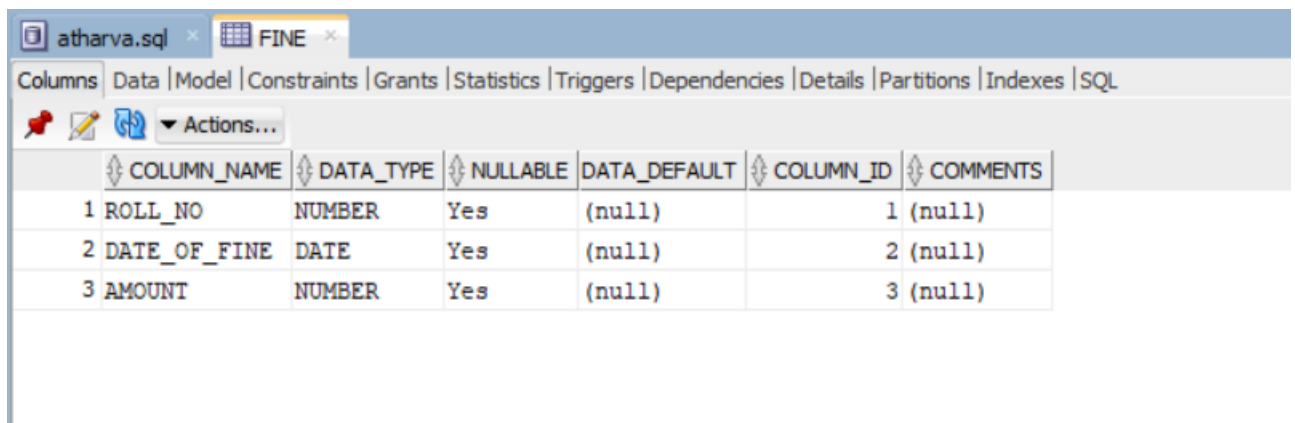
Borrower Table



The screenshot shows the Oracle SQL Developer interface with the 'BORROWER' table selected. The 'Columns' tab is active, displaying the table's structure. The table has five columns: ROLL_NO (NUMBER), NAME (VARCHAR2(20 BYTE)), DATE_OF_ISSUE (DATE), NAME_OF_BOOK (VARCHAR2(20 BYTE)), and STATUS (CHAR(1 BYTE)). All columns are nullable and have a default value of (null).

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ROLL_NO	NUMBER	Yes	(null)	1	(null)
2	NAME	VARCHAR2(20 BYTE)	Yes	(null)	2	(null)
3	DATE_OF_ISSUE	DATE	Yes	(null)	3	(null)
4	NAME_OF_BOOK	VARCHAR2(20 BYTE)	Yes	(null)	4	(null)
5	STATUS	CHAR(1 BYTE)	Yes	(null)	5	(null)

Fine Table



The screenshot shows the Oracle SQL Developer interface with the 'FINE' table selected. The 'Columns' tab is active, displaying the table's structure. The table has three columns: ROLL_NO (NUMBER), DATE_OF_FINE (DATE), and AMOUNT (NUMBER). All columns are nullable and have a default value of (null).

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ROLL_NO	NUMBER	Yes	(null)	1	(null)
2	DATE_OF_FINE	DATE	Yes	(null)	2	(null)
3	AMOUNT	NUMBER	Yes	(null)	3	(null)

PL/SQL Code

```
atharva.sql x BORROWER x
SQL Worksheet History
Worksheet Query Builder

SET SERVEROUTPUT ON

DECLARE
    rno NUMBER;
    nob VARCHAR(20);

    nodays NUMBER;
    doi DATE;
    amount NUMBER(10,2);

BEGIN
    rno := &rno;
    nob := '&nob';

    SELECT DATE_OF_ISSUE INTO doi FROM BORROWER WHERE ROLL_NO = rno AND NAME_OF_BOOK = nob;
    nodays := sysdate - doi;

    IF(nodays >= 15 AND nodays < 30) THEN
        amount := nodays * 5;
    ELSIF(nodays > 30) THEN
        amount := nodays * 50;
    ELSE
        DBMS_OUTPUT.PUT_LINE('NO FINE');
    END IF;

    IF(nodays >= 15) THEN
        INSERT INTO FINE VALUES(rno, sysdate, amount);
    END IF;

    UPDATE BORROWER SET STATUS = 'R' WHERE ROLL_NO = rno AND NAME_OF_BOOK = nob;

EXCEPTION
    WHEN no_data_found THEN
        DBMS_OUTPUT.PUT_LINE('NO RECORD PRESENT');

    WHEN others THEN
        DBMS_OUTPUT.PUT_LINE('ERROR');

END;
```

Output

	ROLL_NO	NAME	DATE_OF_ISSUE	NAME_OF_BOOK	STATUS
1	1	A	10-11-22	C++	R
2	2	B	01-11-22	Java	R
3	3	C	13-10-22	Python	I

atharva.sql × FINE ×

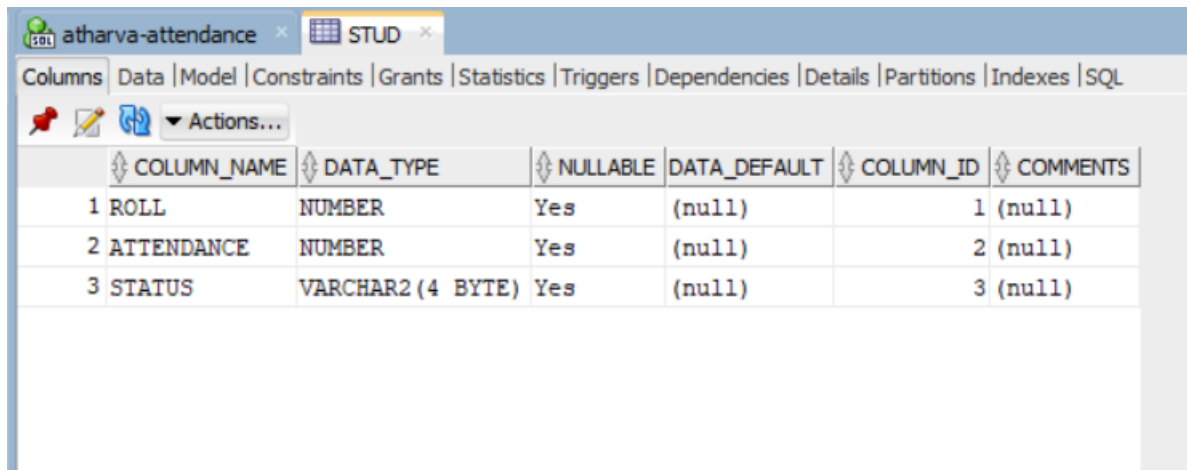
Columns | Data | Model | Constraints | Grants | Statistics | Triggers | Dependencies | De

Sort.. | Filter:

	ROLL_NO	DATE_OF_FINE	AMOUNT
1	2	20-11-22	96.16

(2) Chit 3 –

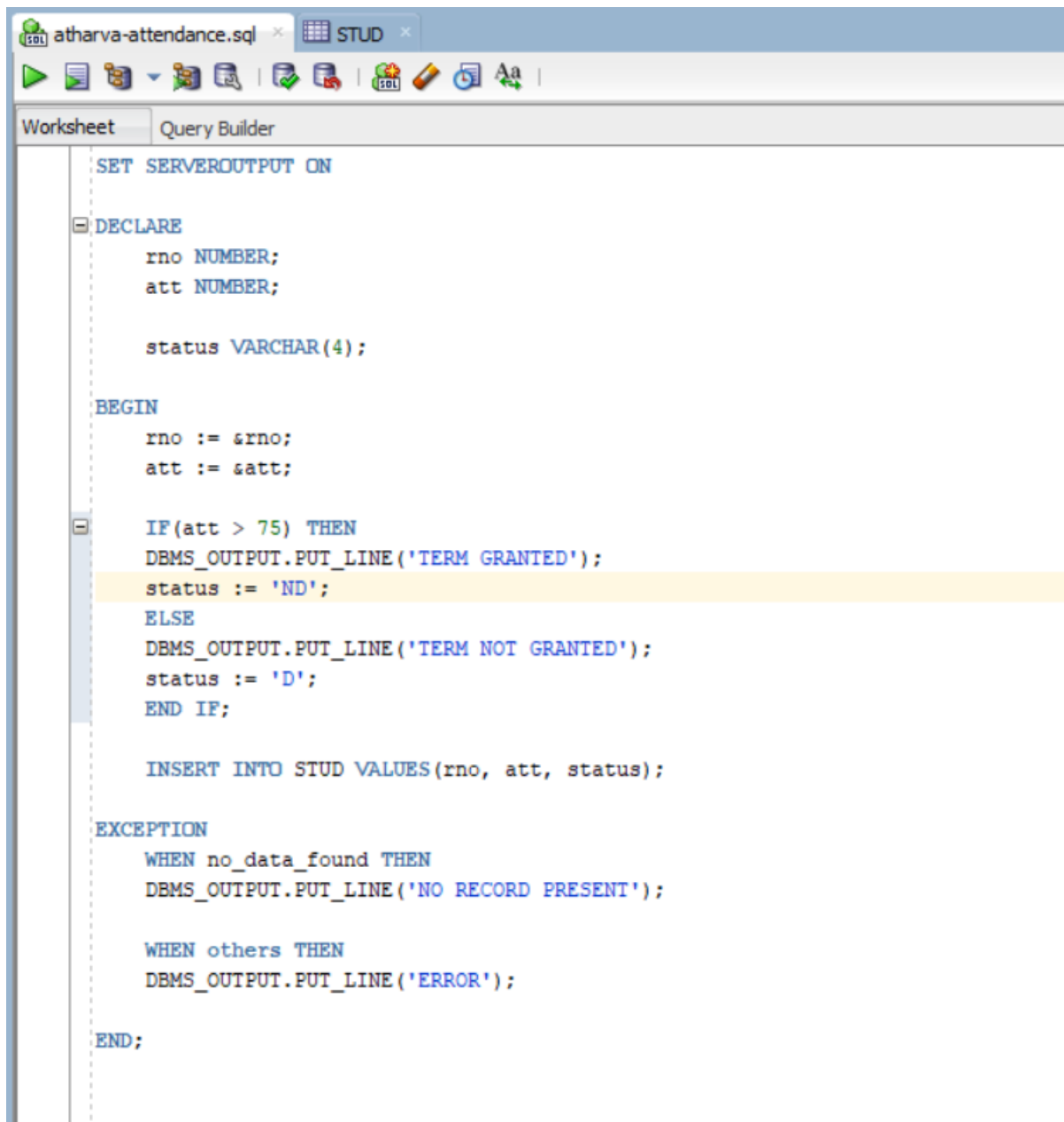
Student Table



The screenshot shows the SQL Developer interface with the 'STUD' table selected. The 'Columns' tab is active, displaying the table's structure. The table has three columns: ROLL, ATTENDANCE, and STATUS. The ROLL and ATTENDANCE columns are of type NUMBER, and the STATUS column is of type VARCHAR2(4 BYTE). All columns are nullable and have a default value of (null). The column IDs are 1, 2, and 3 respectively.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ROLL	NUMBER	Yes	(null)	1	(null)
2	ATTENDANCE	NUMBER	Yes	(null)	2	(null)
3	STATUS	VARCHAR2 (4 BYTE)	Yes	(null)	3	(null)

PL/SQL Code



```
SET SERVEROUTPUT ON

DECLARE
    rno NUMBER;
    att NUMBER;

    status VARCHAR(4);

BEGIN
    rno := &rno;
    att := &att;

    IF(att > 75) THEN
        DBMS_OUTPUT.PUT_LINE('TERM GRANTED');
        status := 'ND';
    ELSE
        DBMS_OUTPUT.PUT_LINE('TERM NOT GRANTED');
        status := 'D';
    END IF;

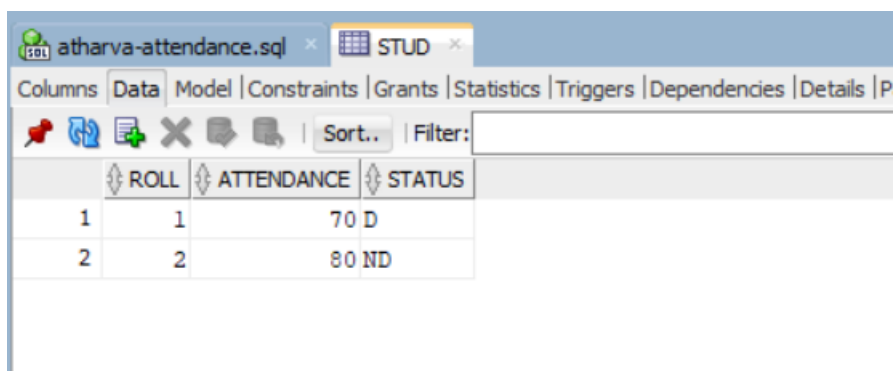
    INSERT INTO STUD VALUES(rno, att, status);

EXCEPTION
    WHEN no_data_found THEN
        DBMS_OUTPUT.PUT_LINE('NO RECORD PRESENT');

    WHEN others THEN
        DBMS_OUTPUT.PUT_LINE('ERROR');

END;
```

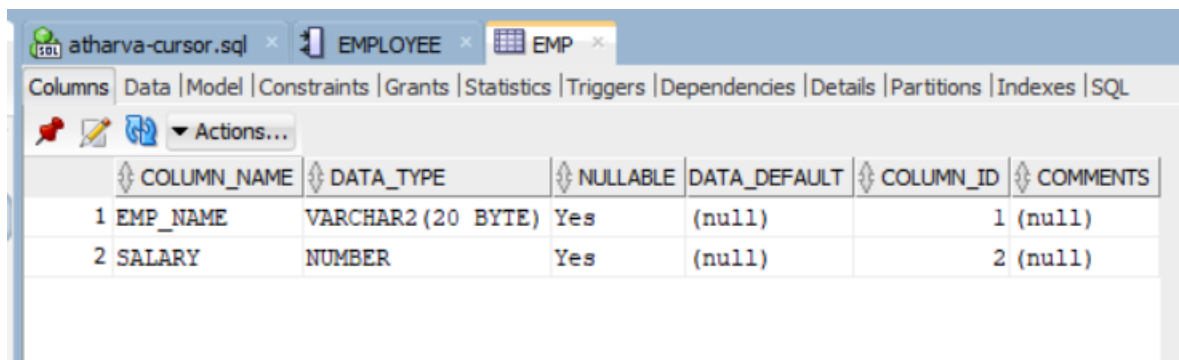
Output



	ROLL	ATTENDANCE	STATUS
1	1	70	D
2	2	80	ND

(3) Chit 5 –

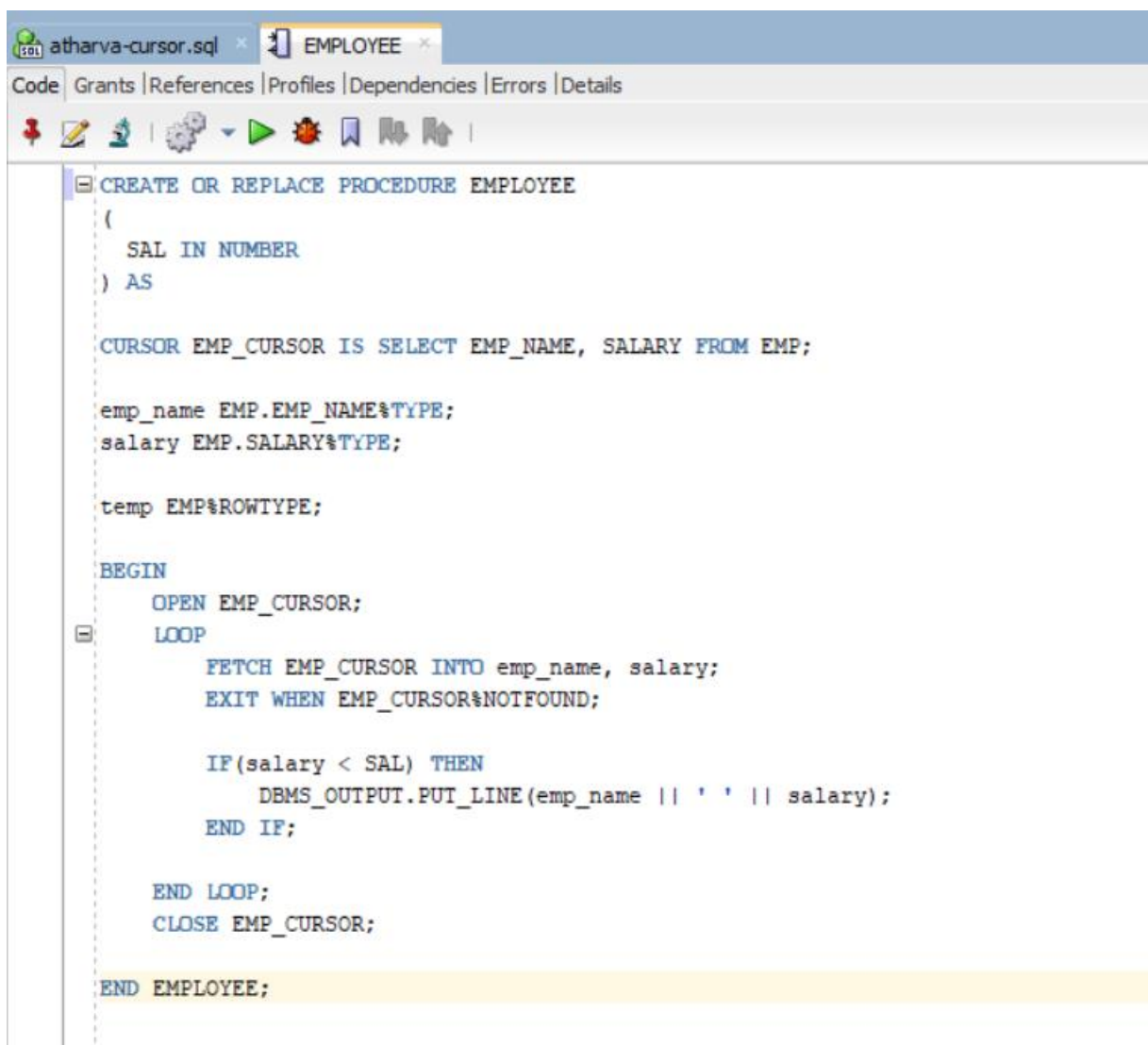
EMP Table



The screenshot shows the SQL Developer interface with the 'EMP' table selected. The 'Columns' tab is active, displaying the table's structure. The table has two columns: EMP_NAME and SALARY.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	EMP_NAME	VARCHAR2(20 BYTE)	Yes	(null)	1	(null)
2	SALARY	NUMBER	Yes	(null)	2	(null)

EMPLOYEE Cursor



The screenshot shows the SQL Developer interface with the 'EMPLOYEE' cursor procedure code. The 'Code' tab is active, displaying the following SQL code:

```
CREATE OR REPLACE PROCEDURE EMPLOYEE
(
    SAL IN NUMBER
) AS

CURSOR EMP_CURSOR IS SELECT EMP_NAME, SALARY FROM EMP;

emp_name EMP.EMP_NAME%TYPE;
salary EMP.SALARY%TYPE;

temp EMP%ROWTYPE;

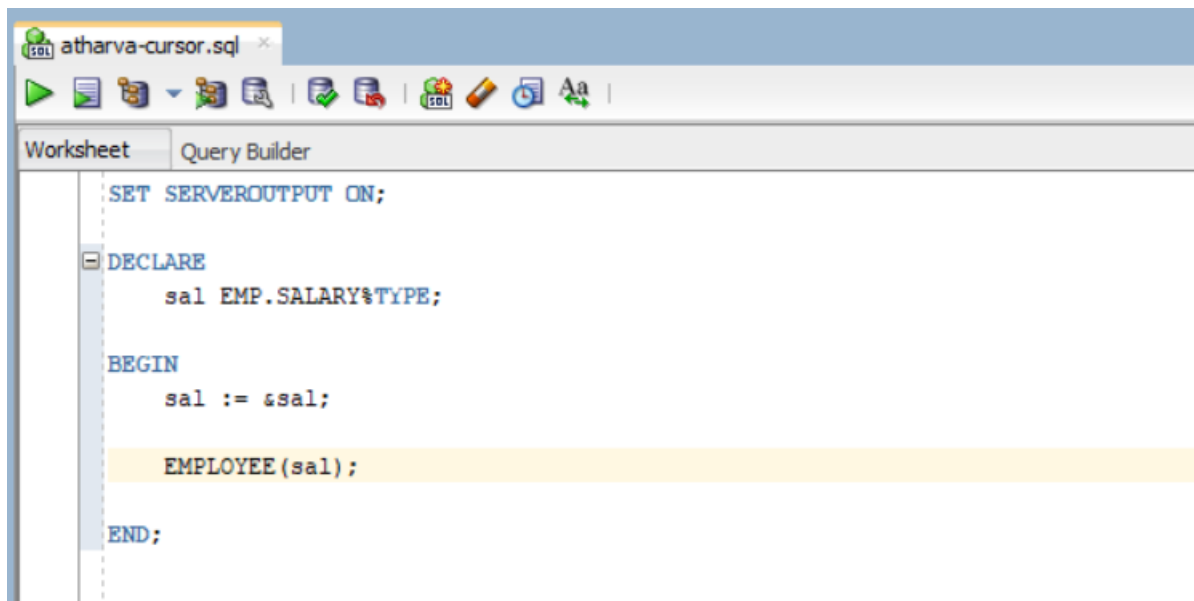
BEGIN
    OPEN EMP_CURSOR;
    LOOP
        FETCH EMP_CURSOR INTO emp_name, salary;
        EXIT WHEN EMP_CURSOR%NOTFOUND;

        IF(salary < SAL) THEN
            DBMS_OUTPUT.PUT_LINE(emp_name || ' ' || salary);
        END IF;

    END LOOP;
    CLOSE EMP_CURSOR;

END EMPLOYEE;
```

PL/SQL Code



The screenshot shows the Oracle SQL Developer interface. The title bar indicates the file is 'atharva-cursor.sql'. The toolbar contains icons for running, saving, undo, redo, and other standard SQL Developer functions. The 'Worksheet' tab is active, displaying the following PL/SQL code:

```
SET SERVEROUTPUT ON;

DECLARE
    sal EMP.SALARY%TYPE;

BEGIN
    sal := &sal;

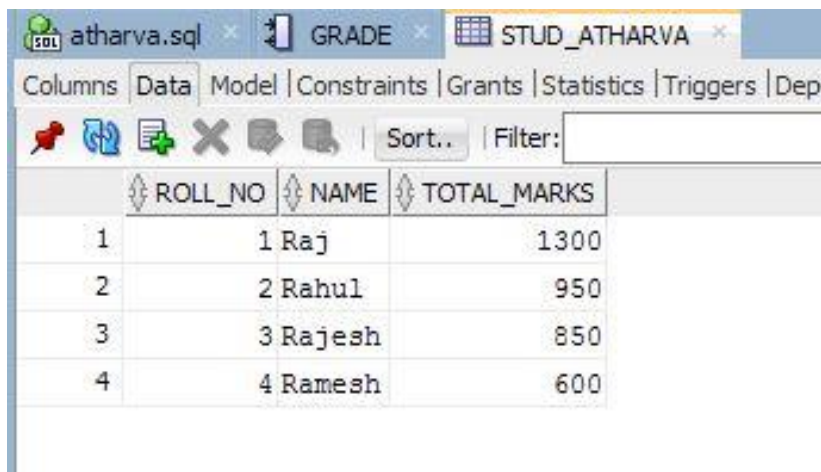
    EMPLOYEE(sal);

END;
```

The line `EMPLOYEE(sal);` is highlighted in yellow.

(4) Chit 6 –

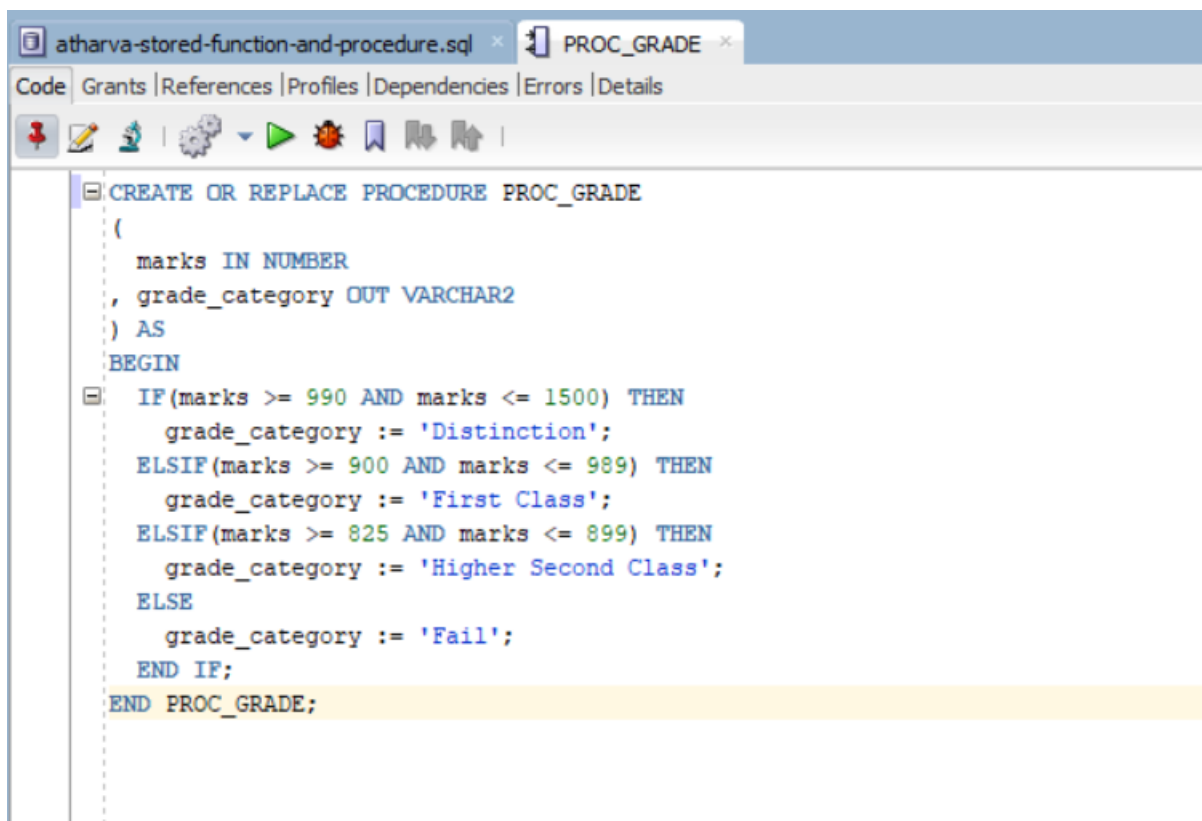
STUDENT Table



The screenshot shows a database interface with a table named 'STUD_ATHARVA'. The table has three columns: 'ROLL_NO', 'NAME', and 'TOTAL_MARKS'. There are four rows of data. The interface includes tabs for 'Columns', 'Data', 'Model', 'Constraints', 'Grants', 'Statistics', 'Triggers', and 'Dep'. A toolbar with various icons and a 'Sort..' button is visible above the table.

ROLL_NO	NAME	TOTAL_MARKS
1	Raj	1300
2	Rahul	950
3	Rajesh	850
4	Ramesh	600

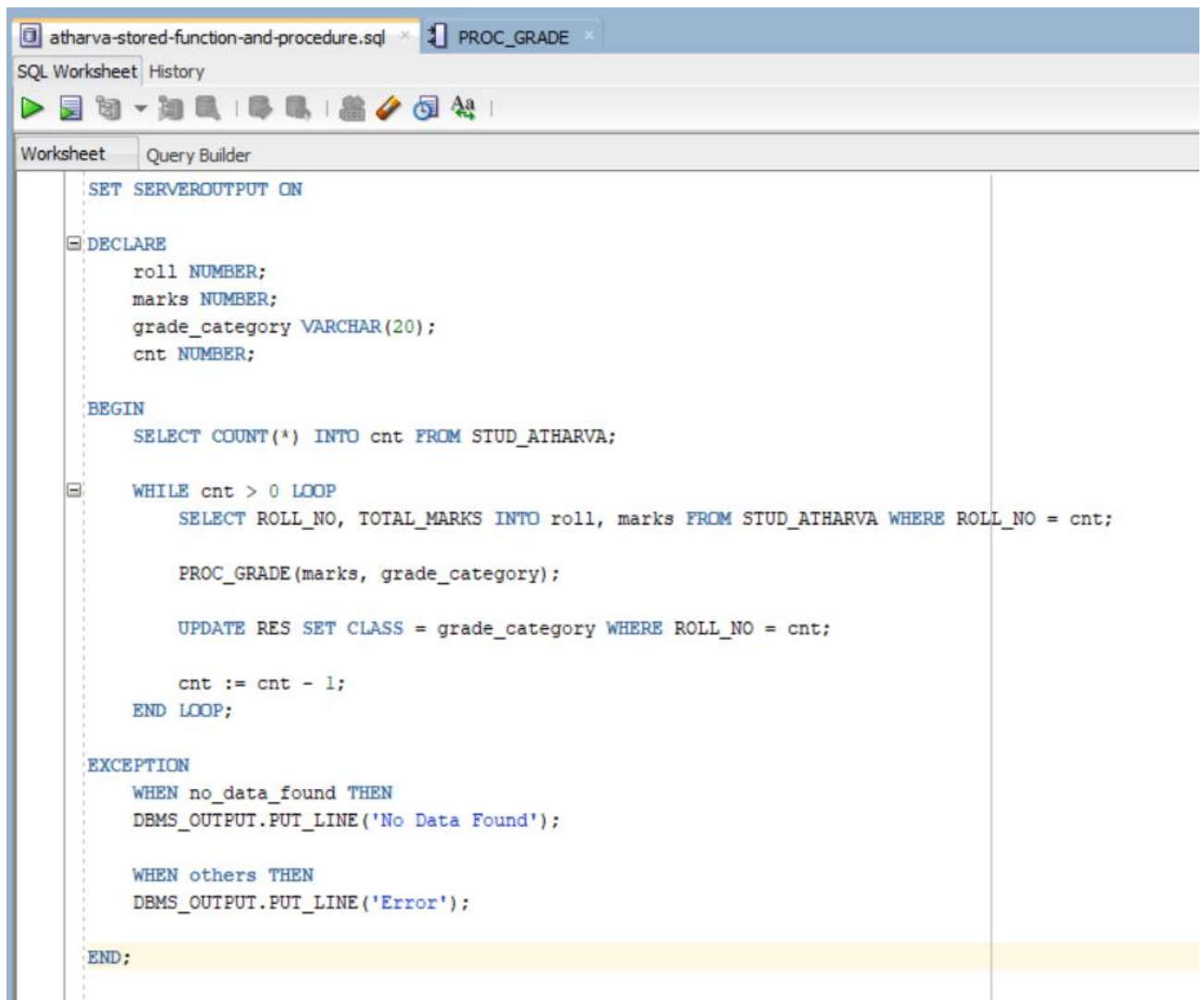
Stored Procedure



The screenshot shows a database interface with a code editor. The code defines a stored procedure named 'PROC_GRADE'. The procedure takes 'marks' as an input parameter of type 'NUMBER' and 'grade_category' as an output parameter of type 'VARCHAR2'. The logic uses an IF-ELSIF-ELSE structure to assign grade categories based on the marks: 'Distinction' for marks >= 990 and <= 1500, 'First Class' for marks >= 900 and <= 989, 'Higher Second Class' for marks >= 825 and <= 899, and 'Fail' for all other cases.

```
CREATE OR REPLACE PROCEDURE PROC_GRADE
(
    marks IN NUMBER
    , grade_category OUT VARCHAR2
) AS
BEGIN
    IF(marks >= 990 AND marks <= 1500) THEN
        grade_category := 'Distinction';
    ELSIF(marks >= 900 AND marks <= 989) THEN
        grade_category := 'First Class';
    ELSIF(marks >= 825 AND marks <= 899) THEN
        grade_category := 'Higher Second Class';
    ELSE
        grade_category := 'Fail';
    END IF;
END PROC_GRADE;
```


PL/SQL Code



```
atharva-stored-function-and-procedure.sql x PROC_GRADE x
SQL Worksheet History
Worksheet Query Builder

SET SERVEROUTPUT ON

DECLARE
    roll NUMBER;
    marks NUMBER;
    grade_category VARCHAR(20);
    cnt NUMBER;

BEGIN
    SELECT COUNT(*) INTO cnt FROM STUD_ATHARVA;

    WHILE cnt > 0 LOOP
        SELECT ROLL_NO, TOTAL_MARKS INTO roll, marks FROM STUD_ATHARVA WHERE ROLL_NO = cnt;

        PROC_GRADE(marks, grade_category);

        UPDATE RES SET CLASS = grade_category WHERE ROLL_NO = cnt;

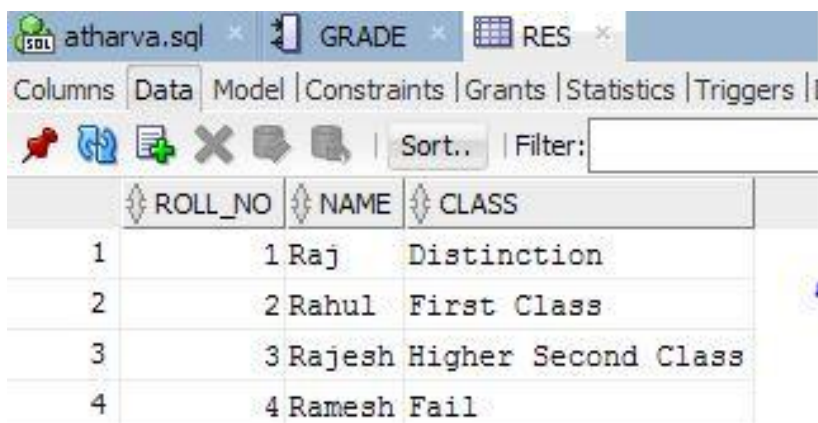
        cnt := cnt - 1;
    END LOOP;

EXCEPTION
    WHEN no_data_found THEN
        DBMS_OUTPUT.PUT_LINE('No Data Found');

    WHEN others THEN
        DBMS_OUTPUT.PUT_LINE('Error');

END;
```

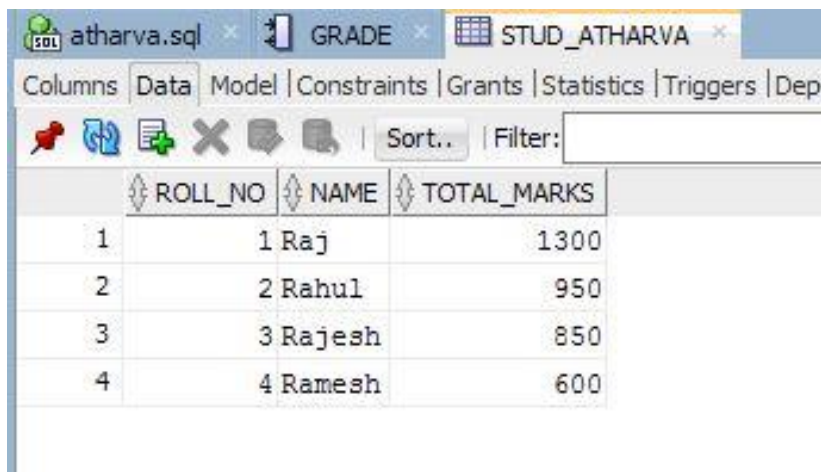
Output



	ROLL_NO	NAME	CLASS
1	1	Raj	Distinction
2	2	Rahul	First Class
3	3	Rajesh	Higher Second Class
4	4	Ramesh	Fail

(5) Chit 7 –

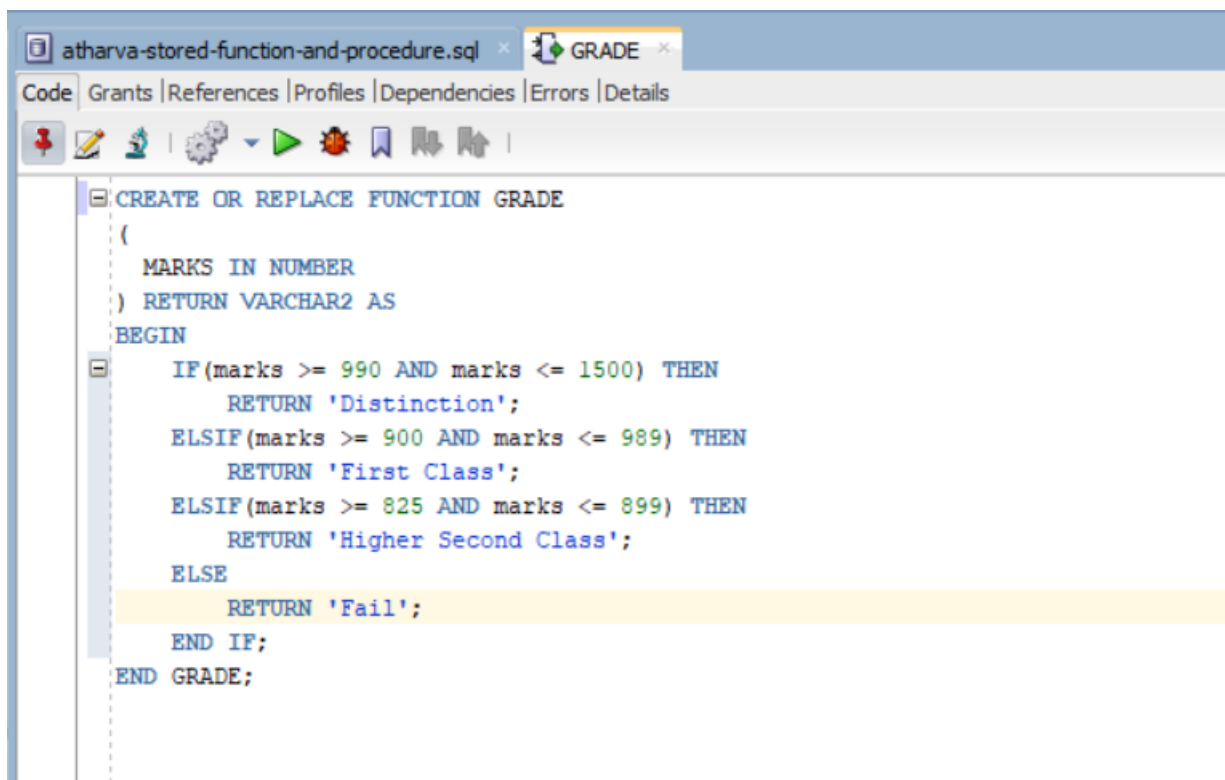
STUDENT Table



The screenshot shows a database interface with a table named 'STUD_ATHARVA'. The table has three columns: 'ROLL_NO', 'NAME', and 'TOTAL_MARKS'. There are four rows of data representing students.

ROLL_NO	NAME	TOTAL_MARKS
1	Raj	1300
2	Rahul	950
3	Rajesh	850
4	Ramesh	600

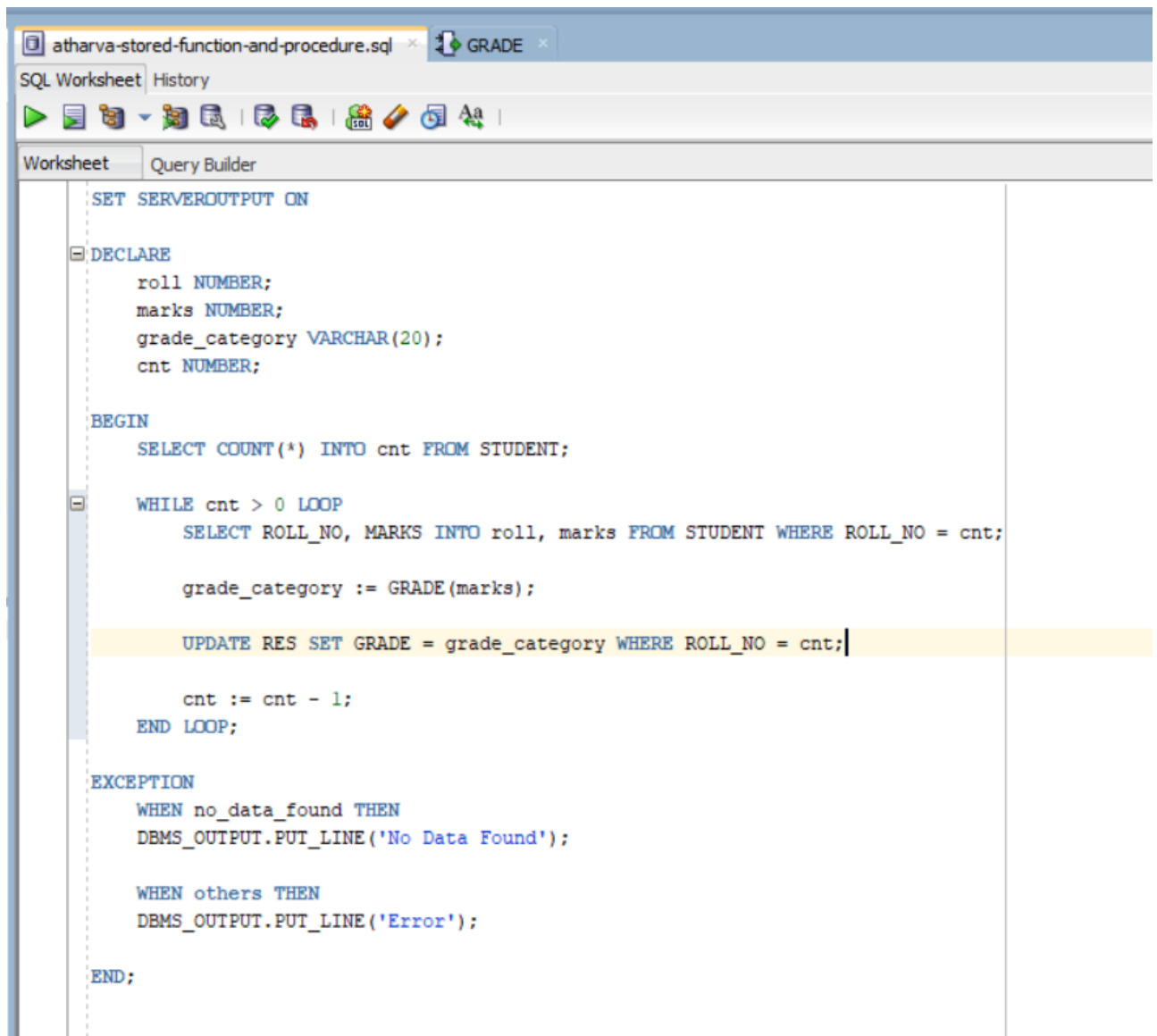
Stored Function



The screenshot shows a database interface with a code editor. The code defines a stored function named 'GRADE' that takes a number of marks as input and returns a grade as a VARCHAR2. The function uses an IF-ELSIF-ELSE structure to determine the grade based on the marks.

```
CREATE OR REPLACE FUNCTION GRADE
(
    MARKS IN NUMBER
) RETURN VARCHAR2 AS
BEGIN
    IF(marks >= 990 AND marks <= 1500) THEN
        RETURN 'Distinction';
    ELSIF(marks >= 900 AND marks <= 989) THEN
        RETURN 'First Class';
    ELSIF(marks >= 825 AND marks <= 899) THEN
        RETURN 'Higher Second Class';
    ELSE
        RETURN 'Fail';
    END IF;
END GRADE;
```

PL/SQL Code



```
atharva-stored-function-and-procedure.sql x GRADE x
SQL Worksheet History
[Icons]
Worksheet Query Builder

SET SERVEROUTPUT ON

DECLARE
    roll NUMBER;
    marks NUMBER;
    grade_category VARCHAR(20);
    cnt NUMBER;

BEGIN
    SELECT COUNT(*) INTO cnt FROM STUDENT;

    WHILE cnt > 0 LOOP
        SELECT ROLL_NO, MARKS INTO roll, marks FROM STUDENT WHERE ROLL_NO = cnt;

        grade_category := GRADE(marks);

        UPDATE RES SET GRADE = grade_category WHERE ROLL_NO = cnt;

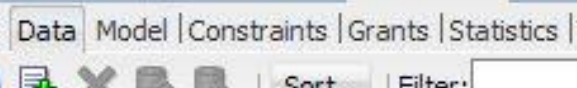
        cnt := cnt - 1;
    END LOOP;

EXCEPTION
    WHEN no_data_found THEN
        DBMS_OUTPUT.PUT_LINE('No Data Found');

    WHEN others THEN
        DBMS_OUTPUT.PUT_LINE('Error');

END;
```

Output

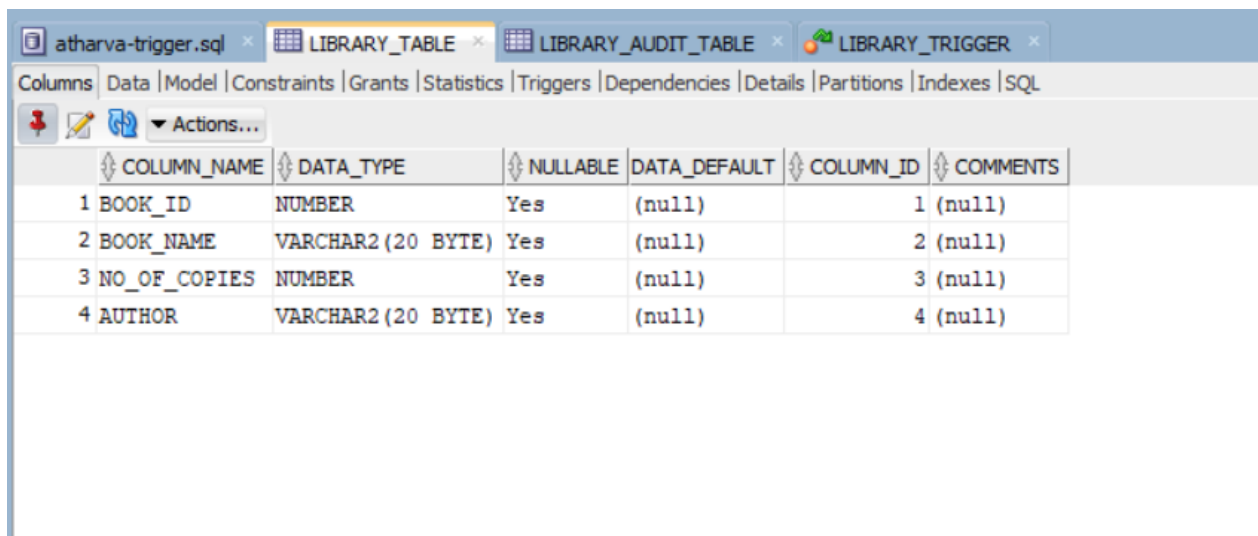


The screenshot shows a SQL Worksheet interface with three tabs: 'atharva.sql', 'GRADE', and 'RES'. The 'RES' tab is active, displaying a table with columns 'ROLL_NO', 'NAME', and 'CLASS'. The table contains four rows of data. The interface includes a top toolbar with icons for saving, undo, redo, and other functions, and a bottom toolbar with a 'Sort..' button and a 'Filter:' input field.

	ROLL_NO	NAME	CLASS
1	1	Raj	Distinction
2	2	Rahul	First Class
3	3	Rajesh	Higher Second Class
4	4	Ramesh	Fail

(6) Chit 11 –

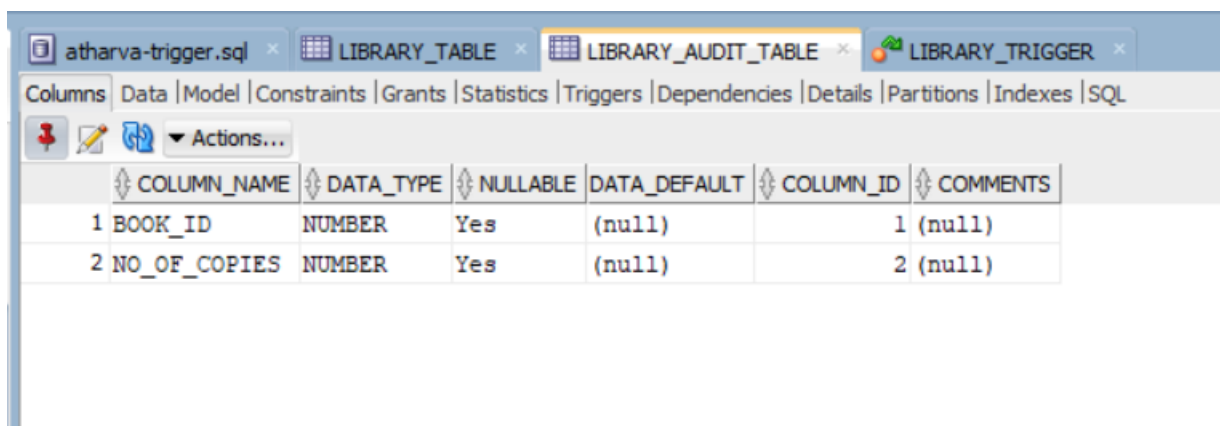
LIBRARY_TABLE



The screenshot shows the Oracle SQL Developer interface with the 'LIBRARY_TABLE' selected. The 'Columns' tab is active, displaying a table with 7 columns: COLUMN_NAME, DATA_TYPE, NULLABLE, DATA_DEFAULT, COLUMN_ID, and COMMENTS. There are 4 rows of data.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BOOK_ID	NUMBER	Yes	(null)	1	(null)
2	BOOK_NAME	VARCHAR2 (20 BYTE)	Yes	(null)	2	(null)
3	NO_OF_COPIES	NUMBER	Yes	(null)	3	(null)
4	AUTHOR	VARCHAR2 (20 BYTE)	Yes	(null)	4	(null)

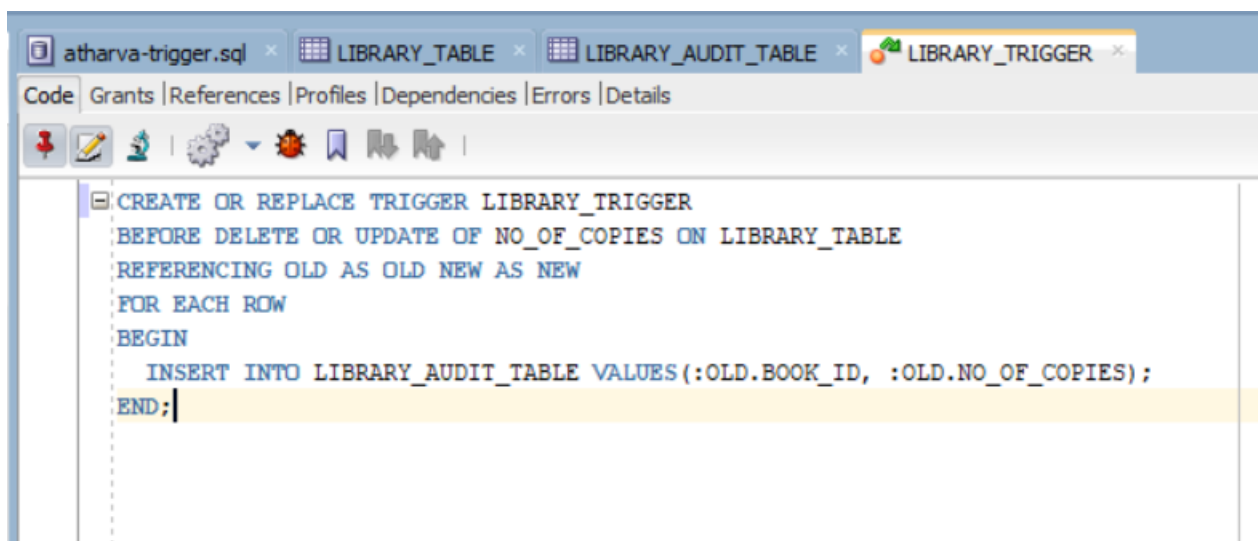
LIBRARY_AUDIT_TABLE



The screenshot shows the Oracle SQL Developer interface with the 'LIBRARY_AUDIT_TABLE' selected. The 'Columns' tab is active, displaying a table with 7 columns: COLUMN_NAME, DATA_TYPE, NULLABLE, DATA_DEFAULT, COLUMN_ID, and COMMENTS. There are 2 rows of data.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BOOK_ID	NUMBER	Yes	(null)	1	(null)
2	NO_OF_COPIES	NUMBER	Yes	(null)	2	(null)

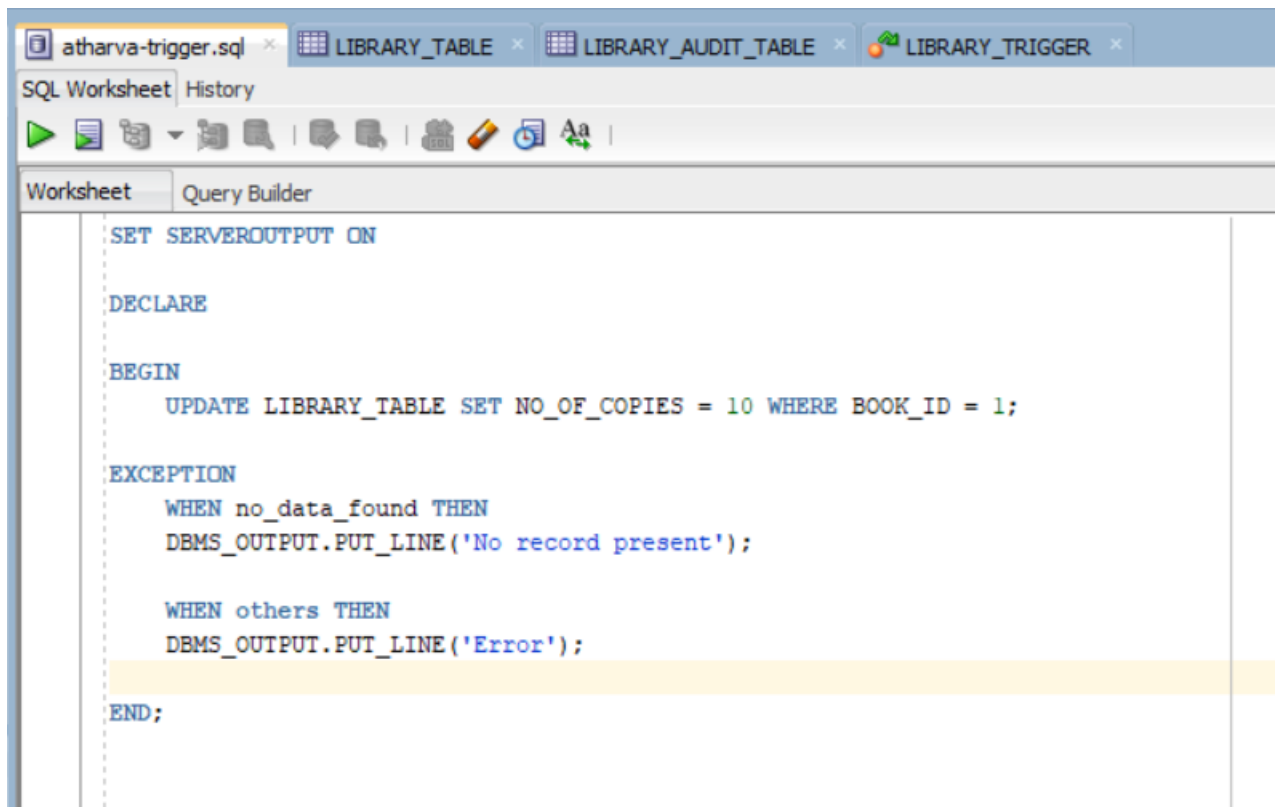
LIBRARY_TRIGGER



The screenshot shows the Oracle SQL Developer interface with the 'LIBRARY_TRIGGER' selected. The 'Code' tab is active, displaying the SQL code for creating or replacing the trigger.

```
CREATE OR REPLACE TRIGGER LIBRARY_TRIGGER
BEFORE DELETE OR UPDATE OF NO_OF_COPIES ON LIBRARY_TABLE
REFERENCING OLD AS OLD NEW AS NEW
FOR EACH ROW
BEGIN
    INSERT INTO LIBRARY_AUDIT_TABLE VALUES (:OLD.BOOK_ID, :OLD.NO_OF_COPIES);
END;
```

PL/SQL CODE



The screenshot shows a SQL Worksheet with the following PL/SQL code:

```
SET SERVEROUTPUT ON

DECLARE

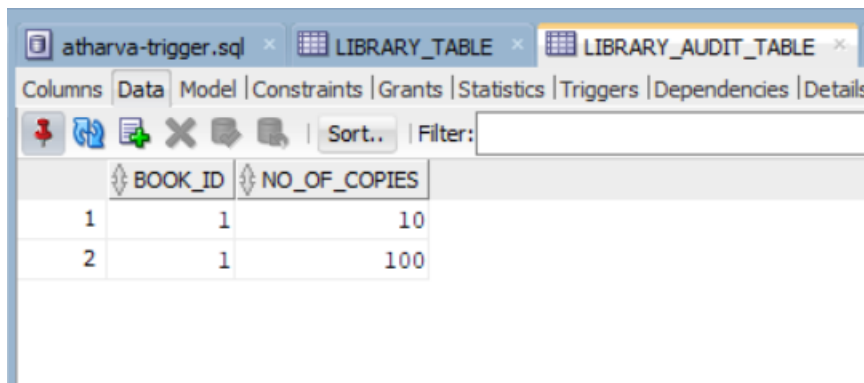
BEGIN
    UPDATE LIBRARY_TABLE SET NO_OF_COPIES = 10 WHERE BOOK_ID = 1;

EXCEPTION
    WHEN no_data_found THEN
        DBMS_OUTPUT.PUT_LINE('No record present');

    WHEN others THEN
        DBMS_OUTPUT.PUT_LINE('Error');

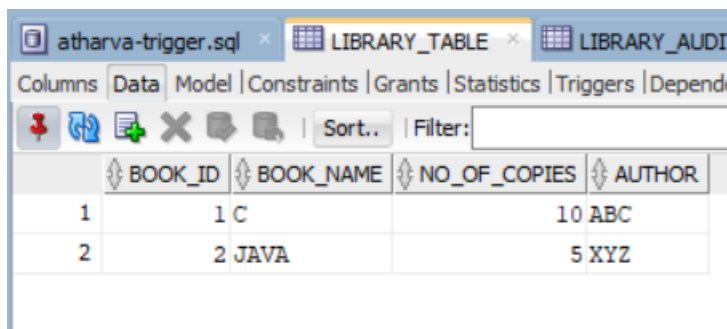
END;
```

Output



The screenshot shows the 'Data' tab of the LIBRARY_TABLE. The table has two columns: BOOK_ID and NO_OF_COPIES. The data is as follows:

BOOK_ID	NO_OF_COPIES
1	10
2	100



The screenshot shows the 'Data' tab of the LIBRARY_AUDIT_TABLE. The table has four columns: BOOK_ID, BOOK_NAME, NO_OF_COPIES, and AUTHOR. The data is as follows:

BOOK_ID	BOOK_NAME	NO_OF_COPIES	AUTHOR
1	1 C	10	ABC
2	2 JAVA	5	XYZ