Ayush Goyal

190905522 CSE D 62

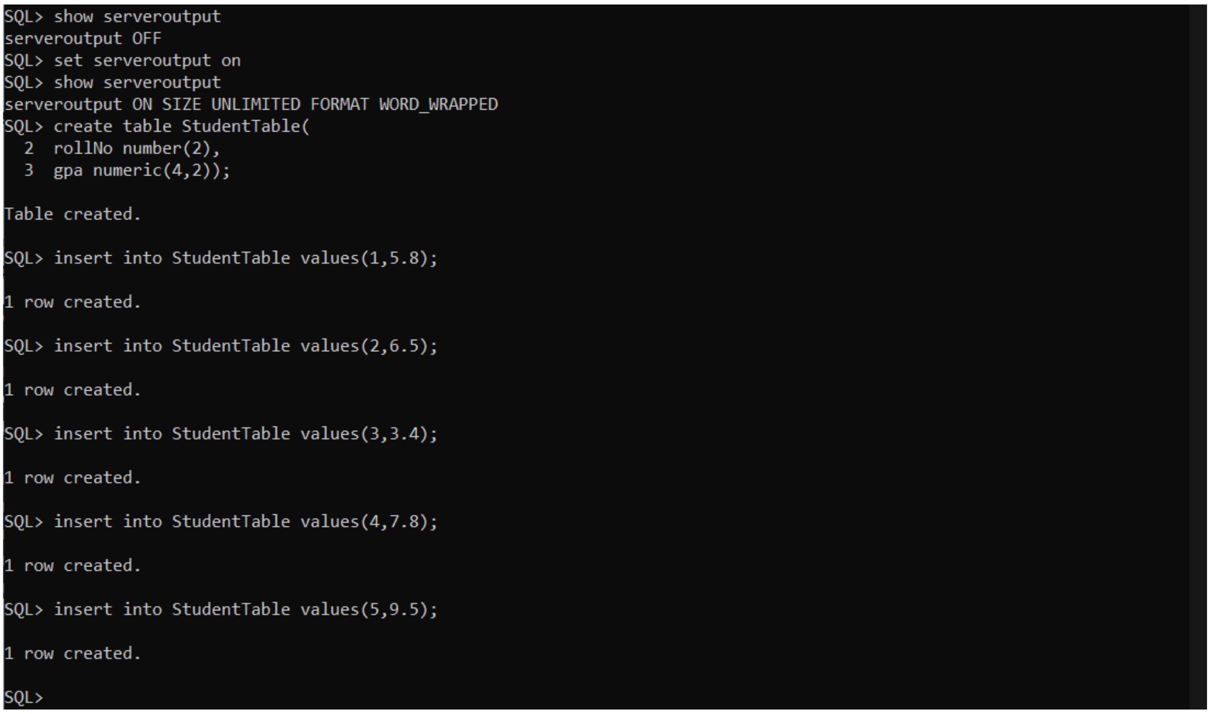
DBS Lab-6 (Week 6) – PL/SQL Basics

Use a table StudentTable(RollNo, GPA) and populate the table with {(1, 5.8); (2, 6.5); (3, 3.4); (4,7.8); (5, 9.5)}unless a different DB schema is explicitly specified.

CODE:

```
set serveroutput on

create table StudentTable(
rollNo number(2),
gpa numeric(4,2));

insert into StudentTable values(1,5.8);
insert into StudentTable values(2,6.5);
insert into StudentTable values(3,3.4);
insert into StudentTable values(4,7.8);
insert into StudentTable values(5,9.5);
```

OUTPUT:



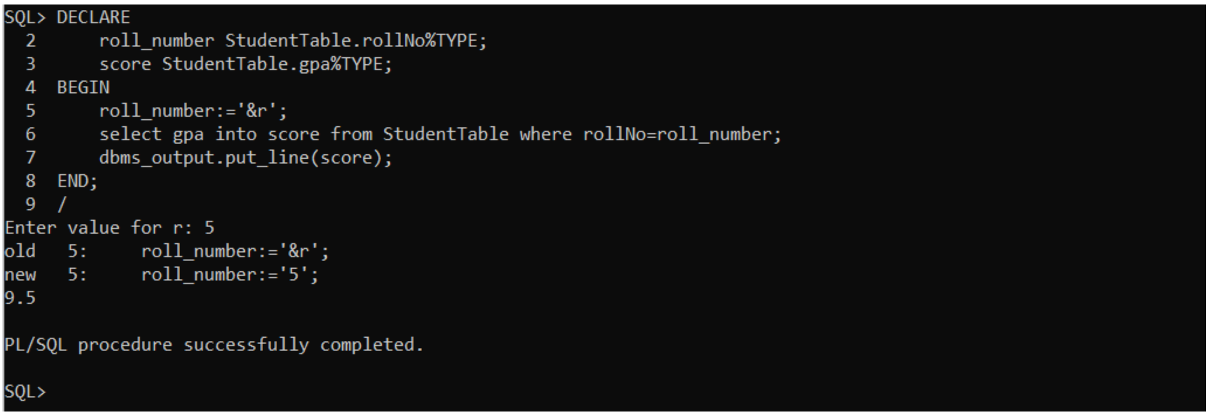
```
SQL> show serveroutput
serveroutput OFF
SQL> set serveroutput on
SQL> show serveroutput
serveroutput ON SIZE UNLIMITED FORMAT WORD_WRAPPED
SQL> create table StudentTable(
  2  rollNo number(2),
  3  gpa numeric(4,2));

Table created.

SQL> insert into StudentTable values(1,5.8);

1 row created.

SQL> insert into StudentTable values(2,6.5);

1 row created.

SQL> insert into StudentTable values(3,3.4);

1 row created.

SQL> insert into StudentTable values(4,7.8);

1 row created.

SQL> insert into StudentTable values(5,9.5);

1 row created.

SQL>
```

1. **Write a PL/SQL block to display the GPA of given student.**

CODE:

```
DECLARE
    roll_number StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
BEGIN
    roll_number:='&r';
    select gpa into score from StudentTable where rollNo=roll_number;
    dbms_output.put_line(score);
END;
/
```

OUTPUT:



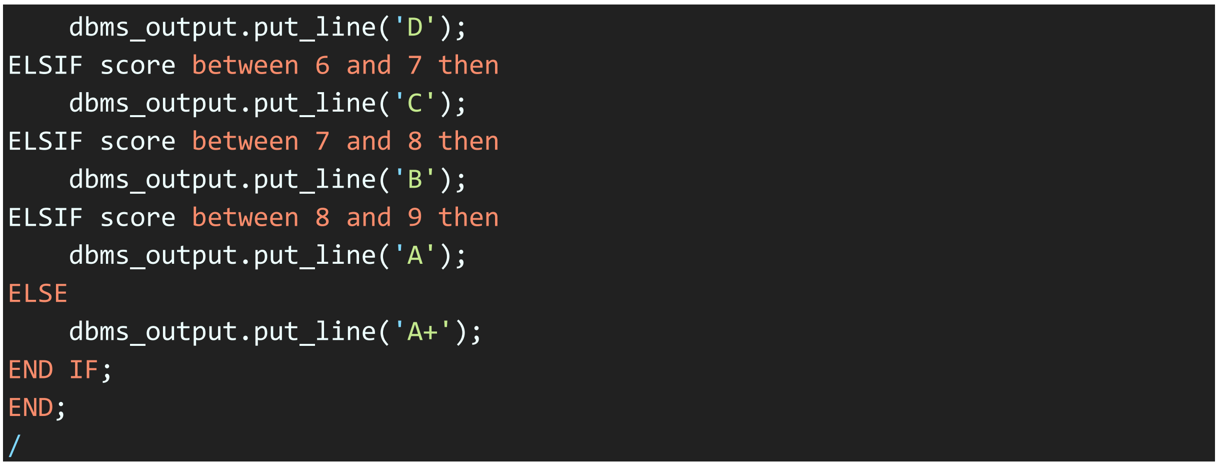
```
SQL> DECLARE
  2      roll_number StudentTable.rollNo%TYPE;
  3      score StudentTable.gpa%TYPE;
  4  BEGIN
  5      roll_number:='&r';
  6      select gpa into score from StudentTable where rollNo=roll_number;
  7      dbms_output.put_line(score);
  8  END;
  9  /
Enter value for r: 5
old   5:     roll_number:='&r';
new   5:     roll_number:='5';
9.5

PL/SQL procedure successfully completed.

SQL>
```

Usage of IF –THEN:

2. **Write a PL/SQL block to display the letter grade (0-4: F; 4-5: E; 5-6: D; 6-7: C; 7-8: B; 8-9: A; 9-10: A+} of given student.**

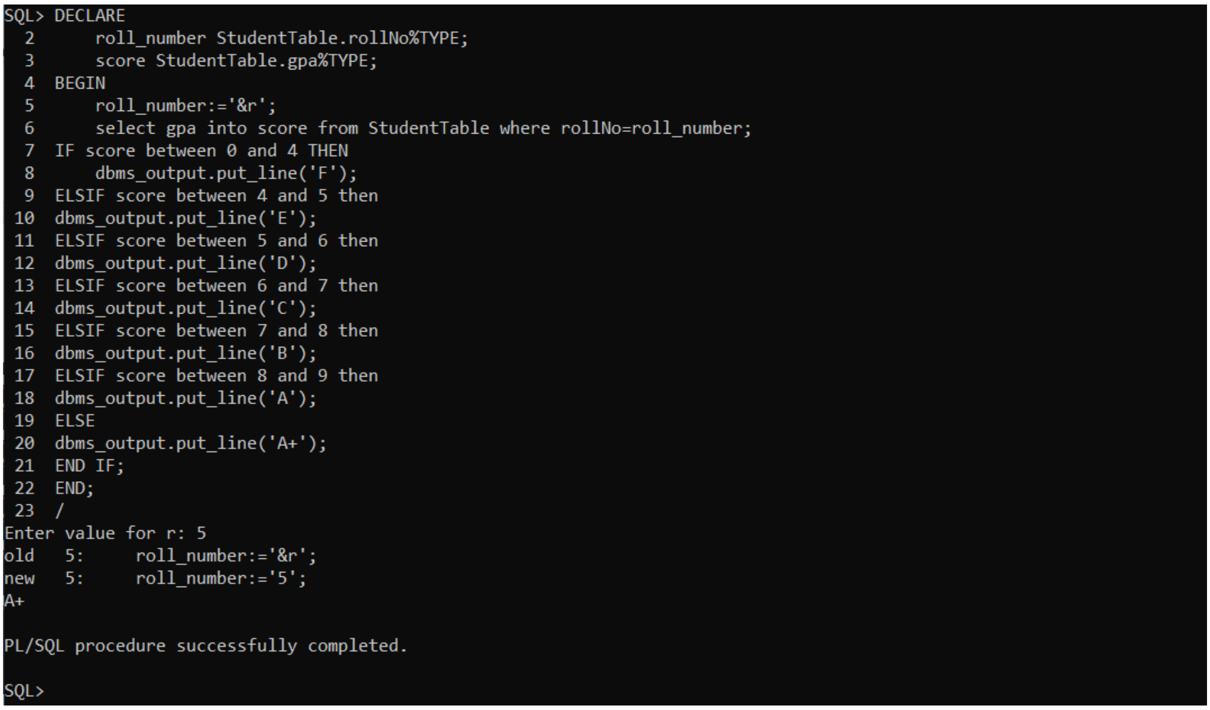
CODE:

```
DECLARE
    roll_number StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
BEGIN
    roll_number:='&r';
    select gpa into score from StudentTable where rollNo=roll_number;
IF score between 0 and 4 THEN
    dbms_output.put_line('F');
ELSIF score between 4 and 5 then
    dbms_output.put_line('E');
ELSIF score between 5 and 6 then
```



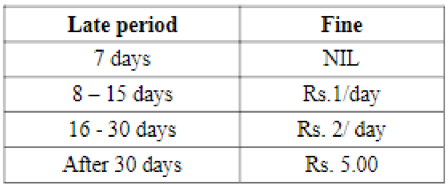
```
    dbms_output.put_line('D');
ELSIF score between 6 and 7 then
    dbms_output.put_line('C');
ELSIF score between 7 and 8 then
    dbms_output.put_line('B');
ELSIF score between 8 and 9 then
    dbms_output.put_line('A');
ELSE
    dbms_output.put_line('A+');
END IF;
END;
/
```

OUTPUT:



```
SQL> DECLARE
  2      roll_number StudentTable.rollNo%TYPE;
  3      score StudentTable.gpa%TYPE;
  4  BEGIN
  5      roll_number:='&r';
  6      select gpa into score from StudentTable where rollNo=roll_number;
  7  IF score between 0 and 4 THEN
  8      dbms_output.put_line('F');
  9  ELSIF score between 4 and 5 then
 10  dbms_output.put_line('E');
 11  ELSIF score between 5 and 6 then
 12  dbms_output.put_line('D');
 13  ELSIF score between 6 and 7 then
 14  dbms_output.put_line('C');
 15  ELSIF score between 7 and 8 then
 16  dbms_output.put_line('B');
 17  ELSIF score between 8 and 9 then
 18  dbms_output.put_line('A');
 19  ELSE
 20  dbms_output.put_line('A+');
 21  END IF;
 22  END;
 23  /
Enter value for r: 5
old   5:      roll_number:='&r';
new   5:      roll_number:='5';
A+

PL/SQL procedure successfully completed.

SQL>
```

3. **Input the date of issue and date of return for a book. Calculate and display the fine with the appropriate message using a PL/SQL block. The fine is charged as per the table 8.1:**



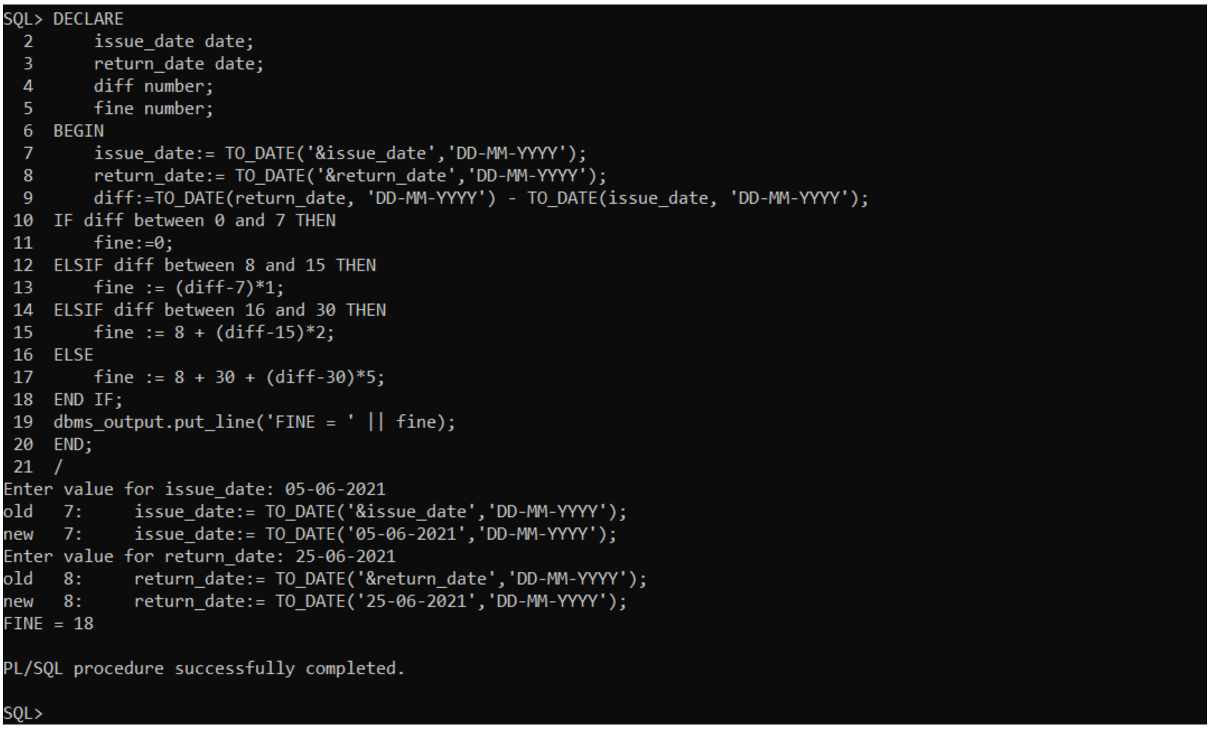
Late period	Fine
7 days	NIL
8 – 15 days	Rs.1/day
16 - 30 days	Rs. 2/ day
After 30 days	Rs. 5.00

Table 8.1

CODE:

```
DECLARE
    issue_date date;
    return_date date;
    diff number;
    fine number;
BEGIN
    issue_date:= TO_DATE('&issue_date','DD-MM-YYYY');
    return_date:= TO_DATE('&return_date','DD-MM-YYYY');
    diff:=TO_DATE(return_date, 'DD-MM-YYYY') - TO_DATE(issue_date, 'DD-MM-YYYY');
IF diff between 0 and 7 THEN
    fine:=0;
ELSIF diff between 8 and 15 THEN
    fine := (diff-7)*1;
ELSIF diff between 16 and 30 THEN
    fine := 8 + (diff-15)*2;
ELSE
    fine := 8 + 30 + (diff-30)*5;
END IF;
dbms_output.put_line('FINE = ' || fine);
END;
/
```

OUTPUT:



```
SQL> DECLARE
  2      issue_date date;
  3      return_date date;
  4      diff number;
  5      fine number;
  6  BEGIN
  7      issue_date:= TO_DATE('&issue_date','DD-MM-YYYY');
  8      return_date:= TO_DATE('&return_date','DD-MM-YYYY');
  9      diff:=TO_DATE(return_date, 'DD-MM-YYYY') - TO_DATE(issue_date, 'DD-MM-YYYY');
 10  IF diff between 0 and 7 THEN
 11      fine:=0;
 12  ELSIF diff between 8 and 15 THEN
 13      fine := (diff-7)*1;
 14  ELSIF diff between 16 and 30 THEN
 15      fine := 8 + (diff-15)*2;
 16  ELSE
 17      fine := 8 + 30 + (diff-30)*5;
 18  END IF;
 19  dbms_output.put_line('FINE = ' || fine);
 20  END;
 21  /
Enter value for issue_date: 05-06-2021
old   7:     issue_date:= TO_DATE('&issue_date','DD-MM-YYYY');
new   7:     issue_date:= TO_DATE('05-06-2021','DD-MM-YYYY');
Enter value for return_date: 25-06-2021
old   8:     return_date:= TO_DATE('&return_date','DD-MM-YYYY');
new   8:     return_date:= TO_DATE('25-06-2021','DD-MM-YYYY');
FINE = 18

PL/SQL procedure successfully completed.

SQL>
```

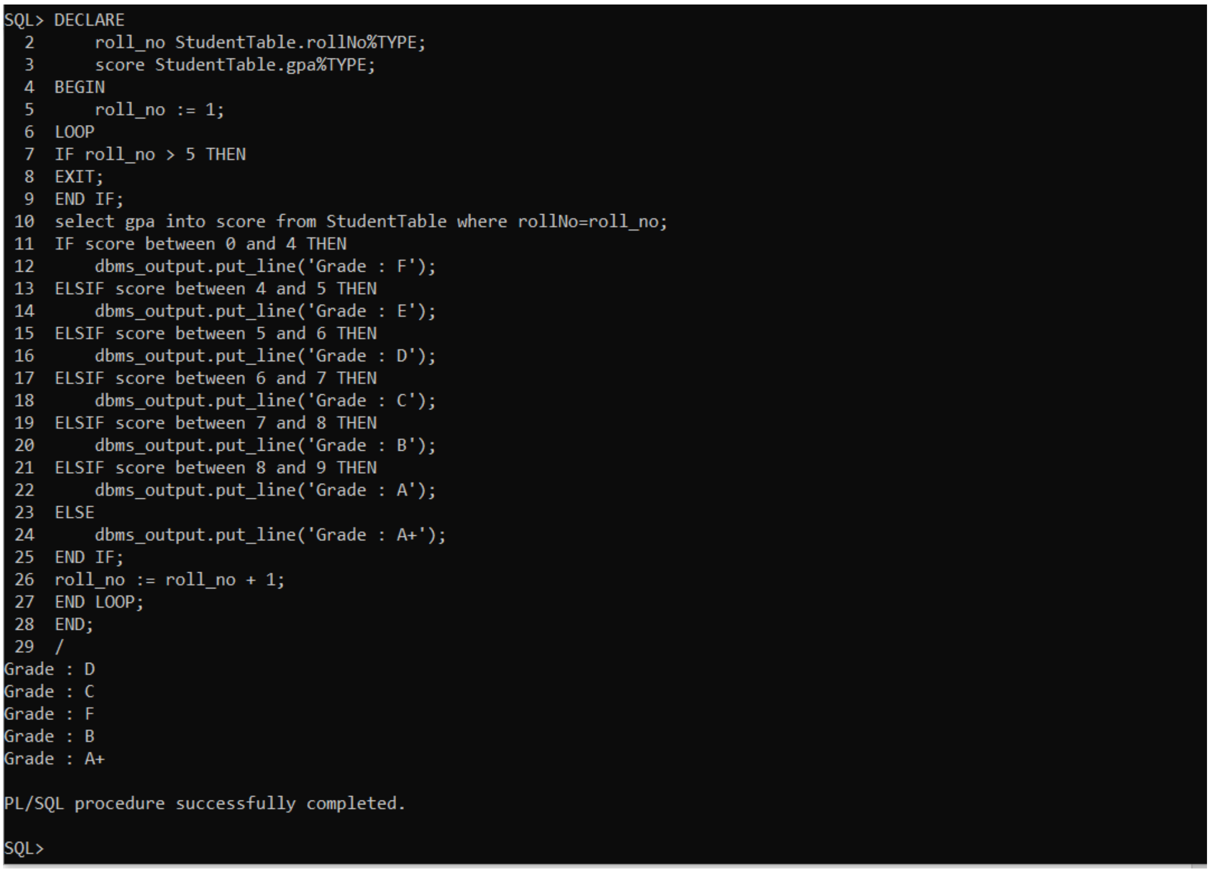
Simple LOOP:

4. **Write a PL/SQL block to print the letter grade of all the students (Roll No: 1 -5).**

CODE:

```
DECLARE
    roll_no StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
BEGIN
    roll_no := 1;
LOOP
IF roll_no > 5 THEN
    EXIT;
END IF;
select gpa into score from StudentTable where rollNo=roll_no;
IF score between 0 and 4 THEN
    dbms_output.put_line('Grade : F');
ELSIF score between 4 and 5 THEN
    dbms_output.put_line('Grade : E');
ELSIF score between 5 and 6 THEN
    dbms_output.put_line('Grade : D');
ELSIF score between 6 and 7 THEN
    dbms_output.put_line('Grade : C');
ELSIF score between 7 and 8 THEN
    dbms_output.put_line('Grade : B');
ELSIF score between 8 and 9 THEN
    dbms_output.put_line('Grade : A');
ELSE
    dbms_output.put_line('Grade : A+');
END IF;
roll_no := roll_no + 1;
END LOOP;
END;
/
```

OUTPUT:



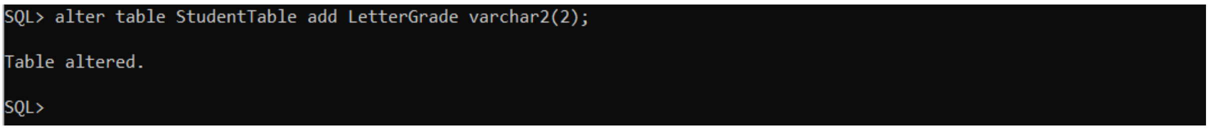
```
SQL> DECLARE
         roll_no StudentTable.rollNo%TYPE;
         score StudentTable.gpa%TYPE;
     BEGIN
         roll_no := 1;
     LOOP
     IF roll_no > 5 THEN
     EXIT;
     END IF;
     select gpa into score from StudentTable where rollNo=roll_no;
     IF score between 0 and 4 THEN
         dbms_output.put_line('Grade : F');
     ELSIF score between 4 and 5 THEN
         dbms_output.put_line('Grade : E');
     ELSIF score between 5 and 6 THEN
         dbms_output.put_line('Grade : D');
     ELSIF score between 6 and 7 THEN
         dbms_output.put_line('Grade : C');
     ELSIF score between 7 and 8 THEN
         dbms_output.put_line('Grade : B');
     ELSIF score between 8 and 9 THEN
         dbms_output.put_line('Grade : A');
     ELSE
         dbms_output.put_line('Grade : A+');
     END IF;
     roll_no := roll_no + 1;
     END LOOP;
     END;
     /
Grade : D
Grade : C
Grade : F
Grade : B
Grade : A+

PL/SQL procedure successfully completed.

SQL>
```

Usage of WHILE:

5. **Alter StudentTable by appending an additional column LetterGrade Varchar2(2). Then write a PL/SQL block to update the table with letter grade of each student.**

CODE:

alter table StudentTable add LetterGrade varchar2(2);

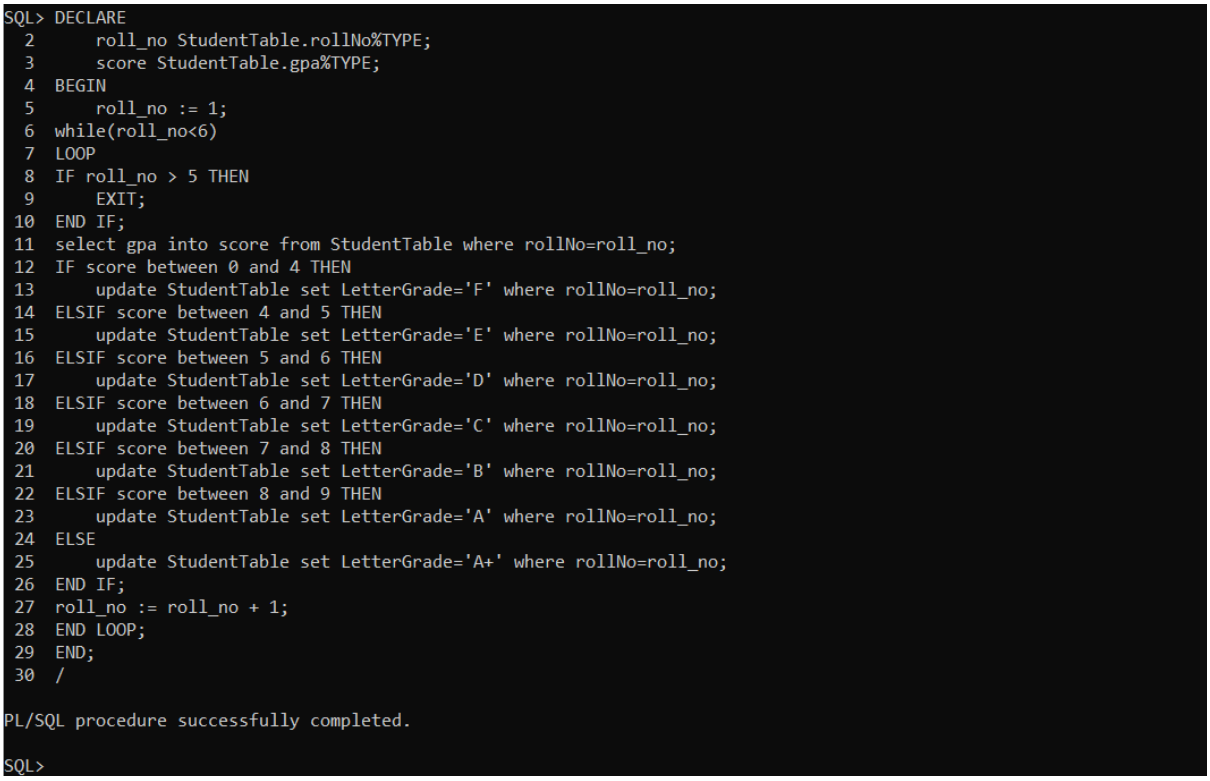


```
SQL> alter table StudentTable add LetterGrade varchar2(2);

Table altered.

SQL>
```

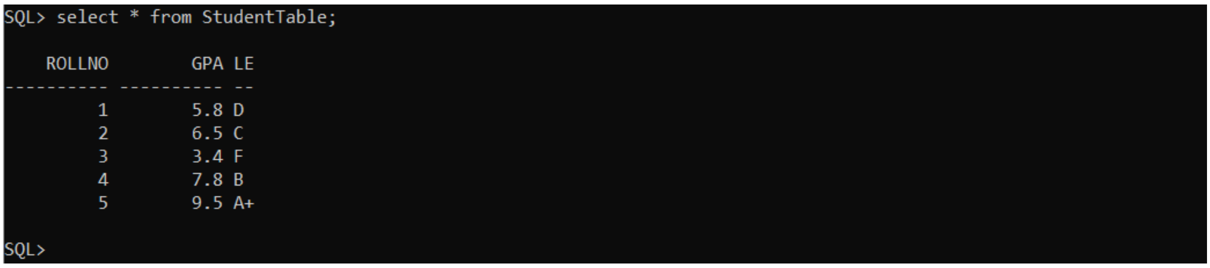
```
DECLARE
    roll_no StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
BEGIN
    roll_no := 1;
while(roll_no<6)
LOOP
IF roll_no > 5 THEN
```

```
    EXIT;
END IF;
select gpa into score from StudentTable where rollNo=roll_no;
IF score between 0 and 4 THEN
    update StudentTable set LetterGrade='F' where rollNo=roll_no;
ELSIF score between 4 and 5 THEN
    update StudentTable set LetterGrade='E' where rollNo=roll_no;
ELSIF score between 5 and 6 THEN
    update StudentTable set LetterGrade='D' where rollNo=roll_no;
ELSIF score between 6 and 7 THEN
    update StudentTable set LetterGrade='C' where rollNo=roll_no;
ELSIF score between 7 and 8 THEN
    update StudentTable set LetterGrade='B' where rollNo=roll_no;
ELSIF score between 8 and 9 THEN
    update StudentTable set LetterGrade='A' where rollNo=roll_no;
ELSE
    update StudentTable set LetterGrade='A+' where rollNo=roll_no;
END IF;
roll_no := roll_no + 1;
END LOOP;
END;
/
```

OUTPUT:



```
SQL> DECLARE
         roll_no StudentTable.rollNo%TYPE;
         score StudentTable.gpa%TYPE;
     BEGIN
         roll_no := 1;
     while(roll_no<6)
     LOOP
     IF roll_no > 5 THEN
         EXIT;
     END IF;
     select gpa into score from StudentTable where rollNo=roll_no;
     IF score between 0 and 4 THEN
         update StudentTable set LetterGrade='F' where rollNo=roll_no;
     ELSIF score between 4 and 5 THEN
         update StudentTable set LetterGrade='E' where rollNo=roll_no;
     ELSIF score between 5 and 6 THEN
         update StudentTable set LetterGrade='D' where rollNo=roll_no;
     ELSIF score between 6 and 7 THEN
         update StudentTable set LetterGrade='C' where rollNo=roll_no;
     ELSIF score between 7 and 8 THEN
         update StudentTable set LetterGrade='B' where rollNo=roll_no;
     ELSIF score between 8 and 9 THEN
         update StudentTable set LetterGrade='A' where rollNo=roll_no;
     ELSE
         update StudentTable set LetterGrade='A+' where rollNo=roll_no;
     END IF;
     roll_no := roll_no + 1;
     END LOOP;
     END;
     /

PL/SQL procedure successfully completed.

SQL>
```



```
SQL> select * from StudentTable;

    ROLLNO        GPA LE
---------- ---------- --
         1        5.8 D
         2        6.5 C
         3        3.4 F
         4        7.8 B
         5        9.5 A+

SQL>
```

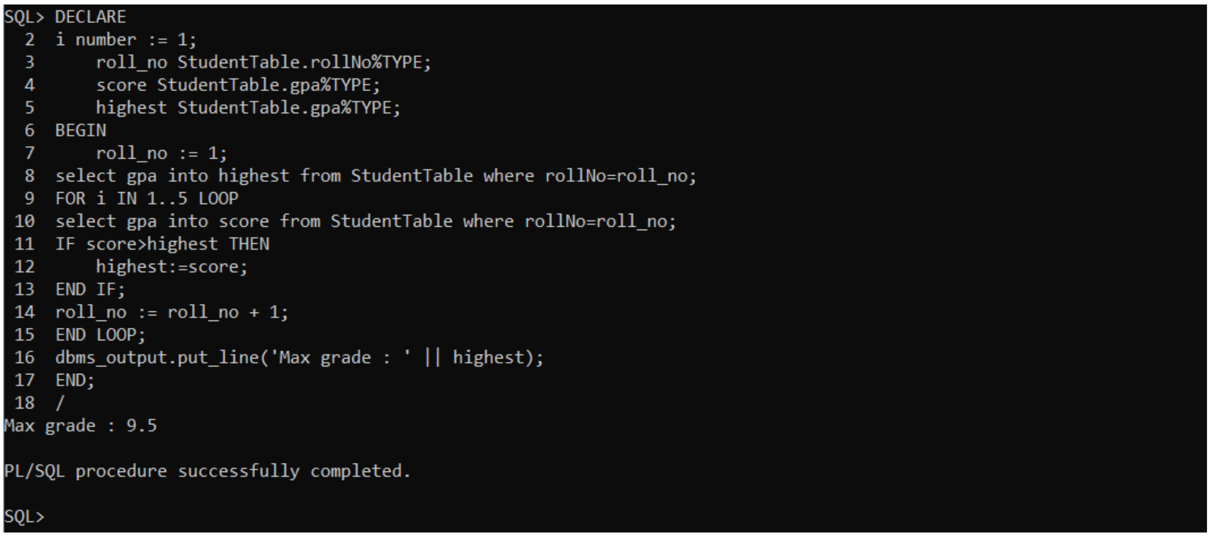
Usage of FOR:

6. **Write a PL/SQL block to find the student with max. GPA without using aggregate function.**

CODE:

```
DECLARE
i number := 1;
    roll_no StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
    highest StudentTable.gpa%TYPE;
BEGIN
    roll_no := 1;
select gpa into highest from StudentTable where rollNo=roll_no;
FOR i IN 1..5 LOOP
select gpa into score from StudentTable where rollNo=roll_no;
IF score>highest THEN
    highest:=score;
END IF;
roll_no := roll_no + 1;
END LOOP;
dbms_output.put_line('Max grade : ' || highest);
END;
/
```

OUTPUT:



```
SQL> DECLARE
     i number := 1;
         roll_no StudentTable.rollNo%TYPE;
         score StudentTable.gpa%TYPE;
         highest StudentTable.gpa%TYPE;
     BEGIN
         roll_no := 1;
     select gpa into highest from StudentTable where rollNo=roll_no;
     FOR i IN 1..5 LOOP
     select gpa into score from StudentTable where rollNo=roll_no;
     IF score>highest THEN
         highest:=score;
     END IF;
     roll_no := roll_no + 1;
     END LOOP;
     dbms_output.put_line('Max grade : ' || highest);
     END;
     /
Max grade : 9.5

PL/SQL procedure successfully completed.

SQL>
```

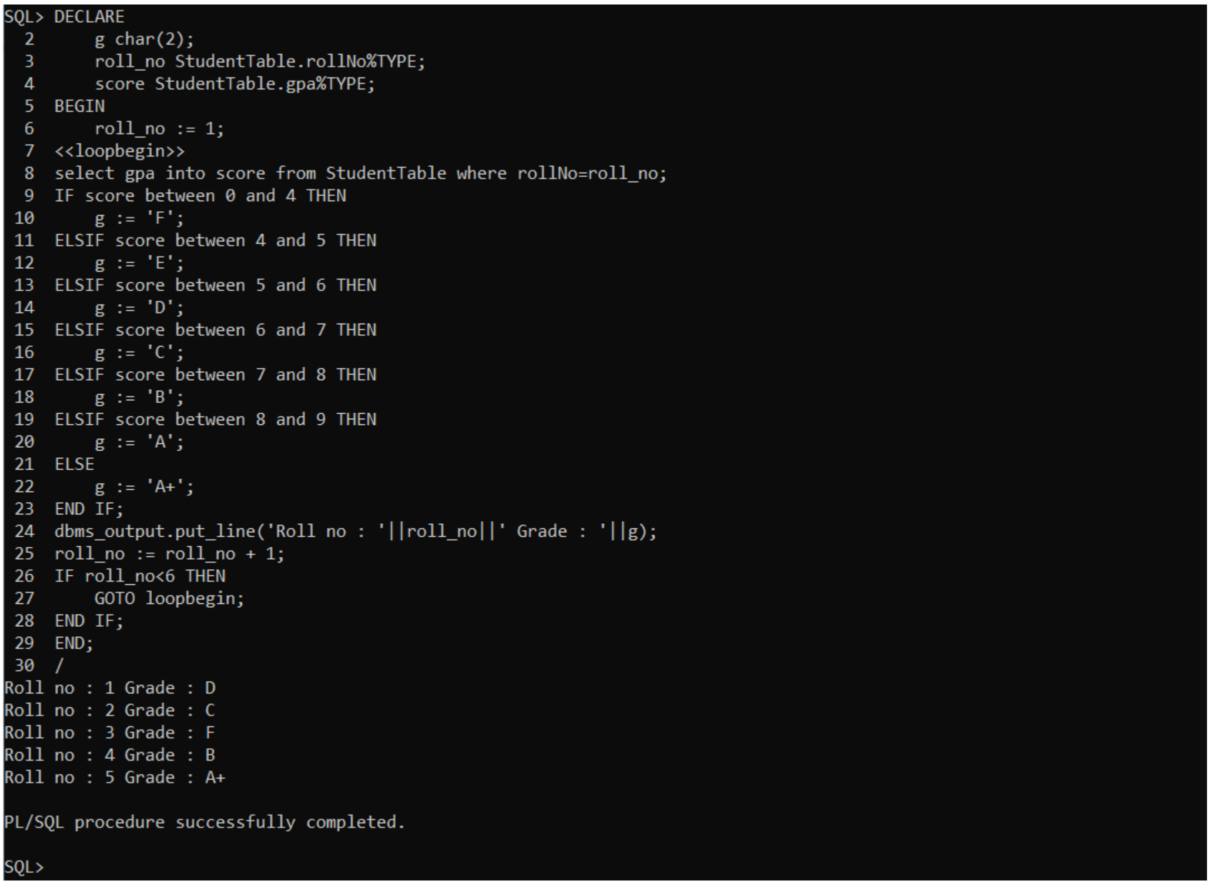
Usage of GOTO:

7. **Implement lab exercise4using GOTO.**

CODE:

```
DECLARE
    g char(2);
    roll_no StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
BEGIN
    roll_no := 1;
<<loopbegin>>
select gpa into score from StudentTable where rollNo=roll_no;
IF score between 0 and 4 THEN
    g := 'F';
ELSIF score between 4 and 5 THEN
    g := 'E';
ELSIF score between 5 and 6 THEN
    g := 'D';
ELSIF score between 6 and 7 THEN
    g := 'C';
ELSIF score between 7 and 8 THEN
    g := 'B';
ELSIF score between 8 and 9 THEN
    g := 'A';
ELSE
    g := 'A+';
END IF;
dbms_output.put_line('Roll no : '||roll_no||' Grade : '||g);
roll_no := roll_no + 1;
IF roll_no<6 THEN
    GOTO loopbegin;
END IF;
END;
/
```

OUTPUT:



```
SQL> DECLARE
         g char(2);
         roll_no StudentTable.rollNo%TYPE;
         score StudentTable.gpa%TYPE;
     BEGIN
         roll_no := 1;
     <<loopbegin>>
     select gpa into score from StudentTable where rollNo=roll_no;
     IF score between 0 and 4 THEN
         g := 'F';
     ELSIF score between 4 and 5 THEN
         g := 'E';
     ELSIF score between 5 and 6 THEN
         g := 'D';
     ELSIF score between 6 and 7 THEN
         g := 'C';
     ELSIF score between 7 and 8 THEN
         g := 'B';
     ELSIF score between 8 and 9 THEN
         g := 'A';
     ELSE
         g := 'A+';
     END IF;
     dbms_output.put_line('Roll no : '||roll_no||' Grade : '||g);
     roll_no := roll_no + 1;
     IF roll_no<6 THEN
         GOTO loopbegin;
     END IF;
     END;
     /
Roll no : 1 Grade : D
Roll no : 2 Grade : C
Roll no : 3 Grade : F
Roll no : 4 Grade : B
Roll no : 5 Grade : A+

PL/SQL procedure successfully completed.

SQL>
```

Exception Handling:

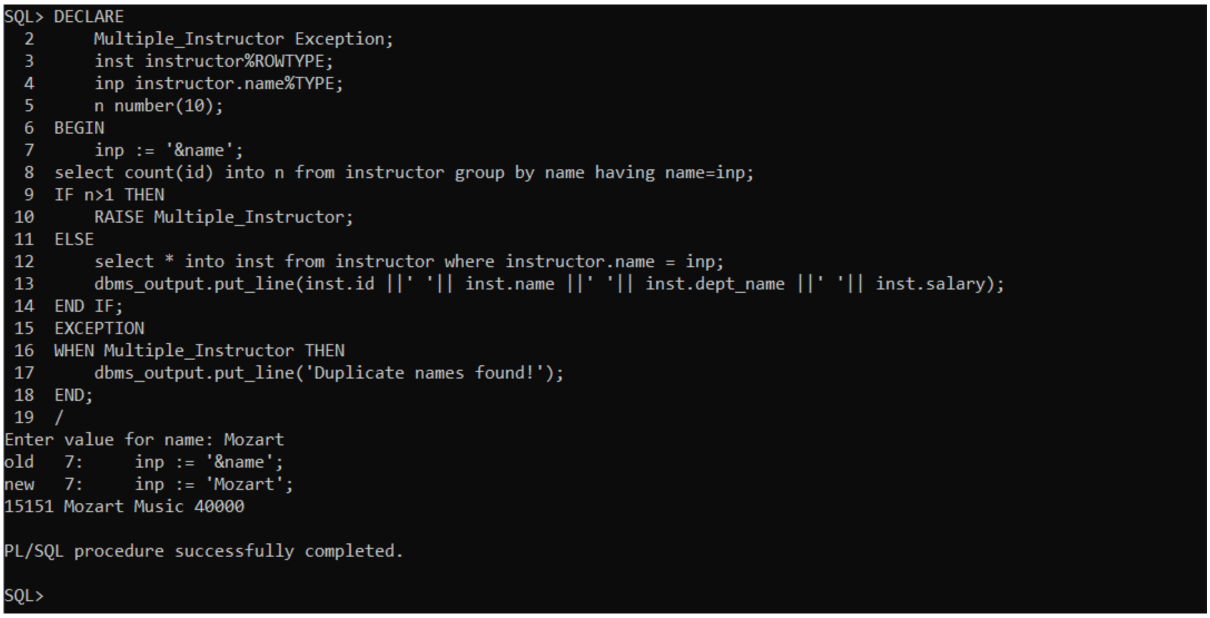
8. **Based on the University database schema, write a PL/SQL block to display the details of the instructor whose name is supplied by the user. Use exceptions to show appropriate error message for the following cases:**

    a. **Multiple instructors with the same name**

CODE:

```
DECLARE
    Multiple_Instructor Exception;
    inst instructor%ROWTYPE;
    inp instructor.name%TYPE;
    n number(10);
BEGIN
    inp := '&name';
select count(id) into n from instructor group by name having name=inp;
IF n>1 THEN
    RAISE Multiple_Instructor;
ELSE
    select * into inst from instructor where instructor.name = inp;
    dbms_output.put_line(inst.id ||' '|| inst.name ||' '|| inst.dept_name ||' '|| inst.salary);
END IF;
EXCEPTION
WHEN Multiple_Instructor THEN
    dbms_output.put_line('Duplicate names found!');
END;
/
```

OUTPUT:



```
SQL> DECLARE
         Multiple_Instructor Exception;
         inst instructor%ROWTYPE;
         inp instructor.name%TYPE;
         n number(10);
     BEGIN
         inp := '&name';
     select count(id) into n from instructor group by name having name=inp;
     IF n>1 THEN
         RAISE Multiple_Instructor;
     ELSE
         select * into inst from instructor where instructor.name = inp;
         dbms_output.put_line(inst.id ||' '|| inst.name ||' '|| inst.dept_name ||' '|| inst.salary);
     END IF;
     EXCEPTION
     WHEN Multiple_Instructor THEN
         dbms_output.put_line('Duplicate names found!');
     END;
     /
Enter value for name: Mozart
old   7:     inp := '&name';
new   7:     inp := 'Mozart';
15151 Mozart Music 40000

PL/SQL procedure successfully completed.

SQL>
```

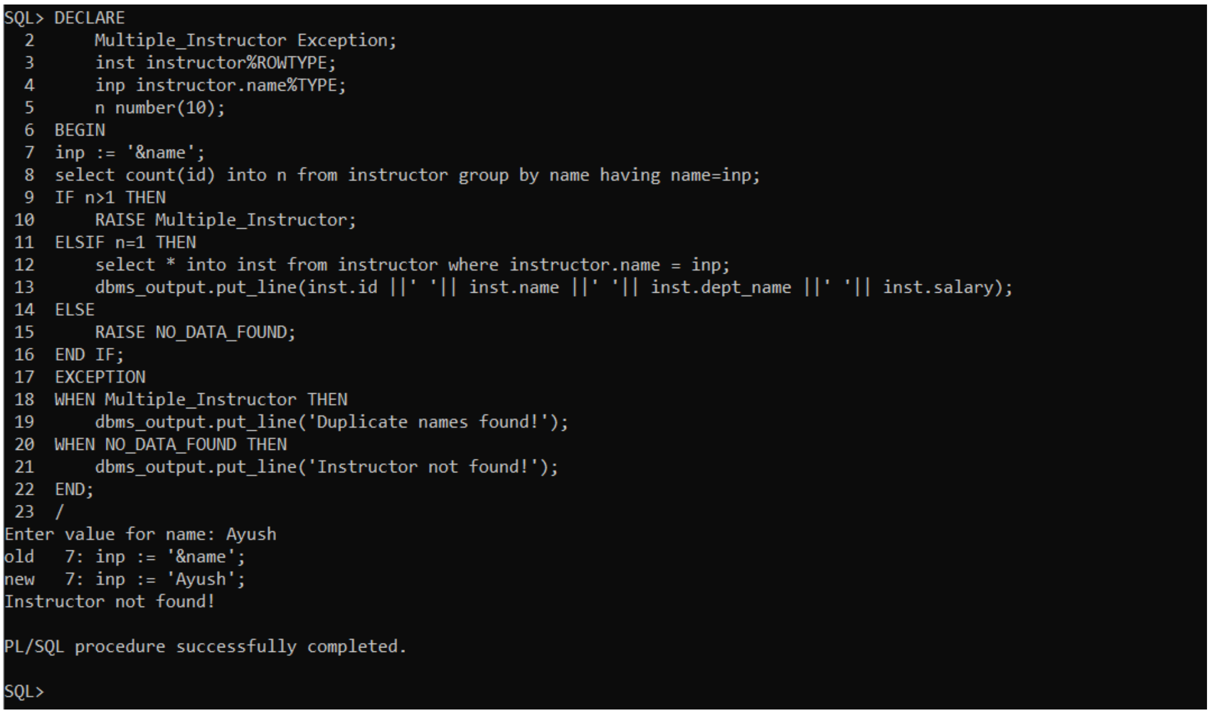
b. **No instructor for the given name**

CODE:

```
DECLARE
    Multiple_Instructor Exception;
    inst instructor%ROWTYPE;
    inp instructor.name%TYPE;
    n number(10);
BEGIN
inp := '&name';
select count(id) into n from instructor group by name having name=inp;
IF n>1 THEN
    RAISE Multiple_Instructor;
ELSIF n=1 THEN
    select * into inst from instructor where instructor.name = inp;
    dbms_output.put_line(inst.id ||' '|| inst.name ||' '|| inst.dept_name ||' '|| inst.salary);
```

```
ELSE
    RAISE NO_DATA_FOUND;
END IF;
EXCEPTION
WHEN Multiple_Instructor THEN
    dbms_output.put_line('Duplicate names found!');
WHEN NO_DATA_FOUND THEN
    dbms_output.put_line('Instructor not found!');
END;
/
```

OUTPUT:

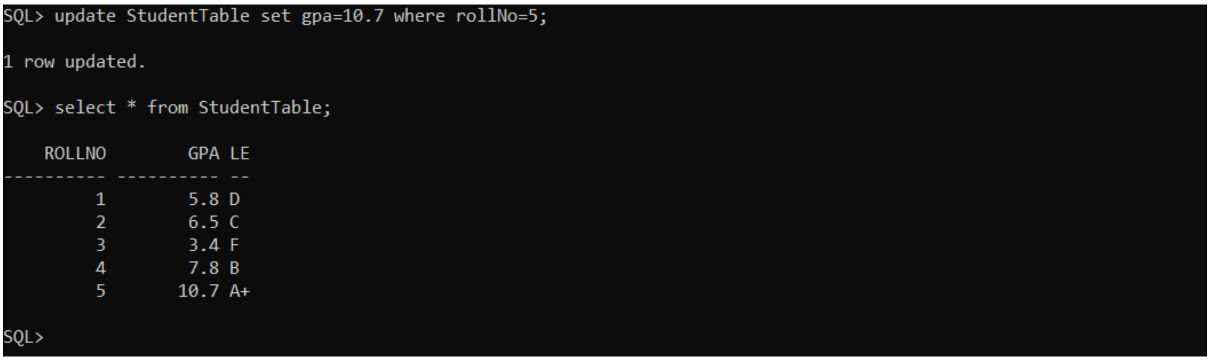


```
SQL> DECLARE
  2      Multiple_Instructor Exception;
  3      inst instructor%ROWTYPE;
  4      inp instructor.name%TYPE;
  5      n number(10);
  6  BEGIN
  7  inp := '&name';
  8  select count(id) into n from instructor group by name having name=inp;
  9  IF n>1 THEN
 10      RAISE Multiple_Instructor;
 11  ELSIF n=1 THEN
 12      select * into inst from instructor where instructor.name = inp;
 13      dbms_output.put_line(inst.id ||' '|| inst.name ||' '|| inst.dept_name ||' '|| inst.salary);
 14  ELSE
 15      RAISE NO_DATA_FOUND;
 16  END IF;
 17  EXCEPTION
 18  WHEN Multiple_Instructor THEN
 19      dbms_output.put_line('Duplicate names found!');
 20  WHEN NO_DATA_FOUND THEN
 21      dbms_output.put_line('Instructor not found!');
 22  END;
 23  /
Enter value for name: Ayush
old   7: inp := '&name';
new   7: inp := 'Ayush';
Instructor not found!

PL/SQL procedure successfully completed.

SQL>
```

9. **Extend lab exercise5 to validate the GPA value used to find letter grade. If it is outside the range, 0 –10, display an error message, 'Out of Range' via an exception handler.**

We need to update someone's gpa to more than 10 to check for exception.

```
update StudentTable set gpa=10.7 where rollNo=5;
```

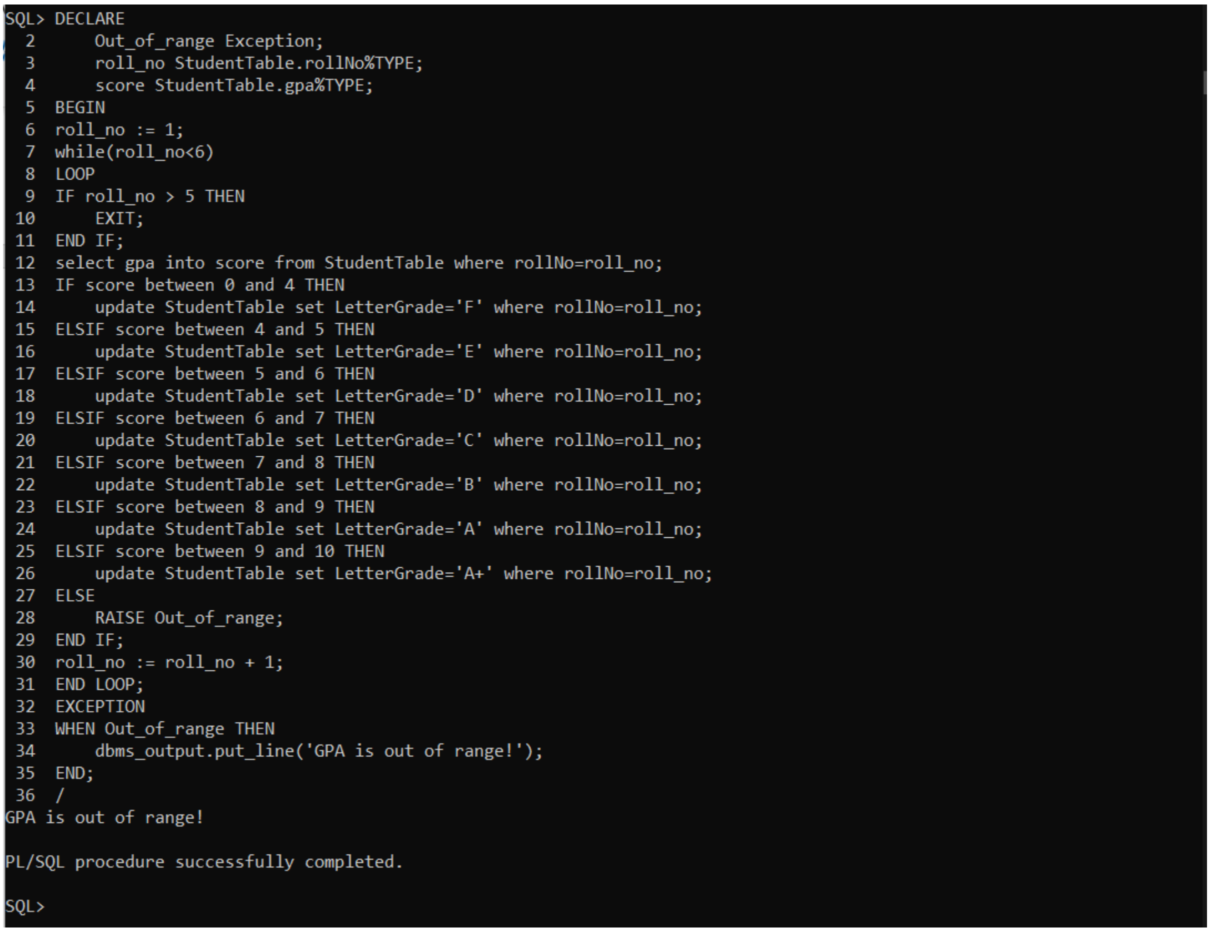


```
SQL> update StudentTable set gpa=10.7 where rollNo=5;

1 row updated.

SQL> select * from StudentTable;

    ROLLNO        GPA LE
---------- ---------- --
         1        5.8 D
         2        6.5 C
         3        3.4 F
         4        7.8 B
         5       10.7 A+

SQL>
```

CODE:

```
DECLARE
    Out_of_range Exception;
    roll_no StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
BEGIN
roll_no := 1;
while(roll_no<6)
LOOP
IF roll_no > 5 THEN
    EXIT;
END IF;
select gpa into score from StudentTable where rollNo=roll_no;
IF score between 0 and 4 THEN
    update StudentTable set LetterGrade='F' where rollNo=roll_no;
ELSIF score between 4 and 5 THEN
    update StudentTable set LetterGrade='E' where rollNo=roll_no;
ELSIF score between 5 and 6 THEN
    update StudentTable set LetterGrade='D' where rollNo=roll_no;
ELSIF score between 6 and 7 THEN
    update StudentTable set LetterGrade='C' where rollNo=roll_no;
ELSIF score between 7 and 8 THEN
    update StudentTable set LetterGrade='B' where rollNo=roll_no;
ELSIF score between 8 and 9 THEN
    update StudentTable set LetterGrade='A' where rollNo=roll_no;
ELSIF score between 9 and 10 THEN
    update StudentTable set LetterGrade='A+' where rollNo=roll_no;
ELSE
    RAISE Out_of_range;
END IF;
roll_no := roll_no + 1;
END LOOP;
EXCEPTION
WHEN Out_of_range THEN
    dbms_output.put_line('GPA is out of range!');
END;
/
```

OUTPUT:



```
SQL> DECLARE
    Out_of_range Exception;
    roll_no StudentTable.rollNo%TYPE;
    score StudentTable.gpa%TYPE;
BEGIN
roll_no := 1;
while(roll_no<6)
LOOP
IF roll_no > 5 THEN
    EXIT;
END IF;
select gpa into score from StudentTable where rollNo=roll_no;
IF score between 0 and 4 THEN
    update StudentTable set LetterGrade='F' where rollNo=roll_no;
ELSIF score between 4 and 5 THEN
    update StudentTable set LetterGrade='E' where rollNo=roll_no;
ELSIF score between 5 and 6 THEN
    update StudentTable set LetterGrade='D' where rollNo=roll_no;
ELSIF score between 6 and 7 THEN
    update StudentTable set LetterGrade='C' where rollNo=roll_no;
ELSIF score between 7 and 8 THEN
    update StudentTable set LetterGrade='B' where rollNo=roll_no;
ELSIF score between 8 and 9 THEN
    update StudentTable set LetterGrade='A' where rollNo=roll_no;
ELSIF score between 9 and 10 THEN
    update StudentTable set LetterGrade='A+' where rollNo=roll_no;
ELSE
    RAISE Out_of_range;
END IF;
roll_no := roll_no + 1;
END LOOP;
EXCEPTION
WHEN Out_of_range THEN
    dbms_output.put_line('GPA is out of range!');
END;
/
GPA is out of range!

PL/SQL procedure successfully completed.

SQL>
```

THE END