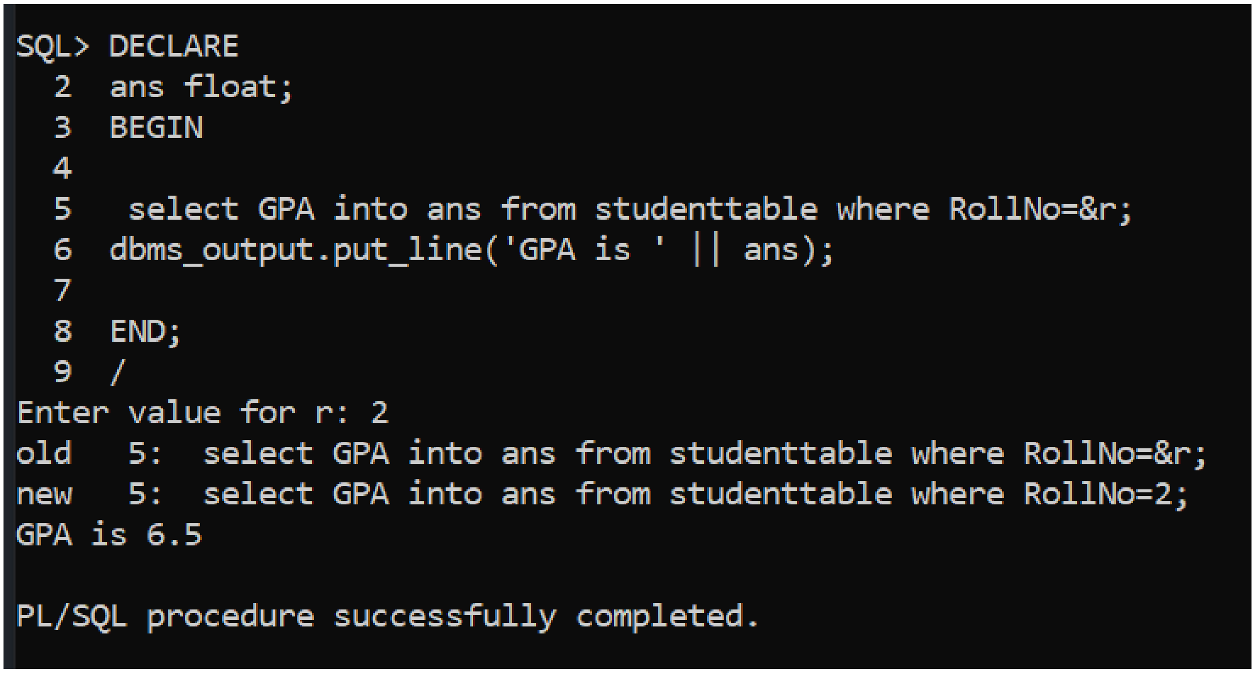
Parth Shukla
190905104
Lab 6

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

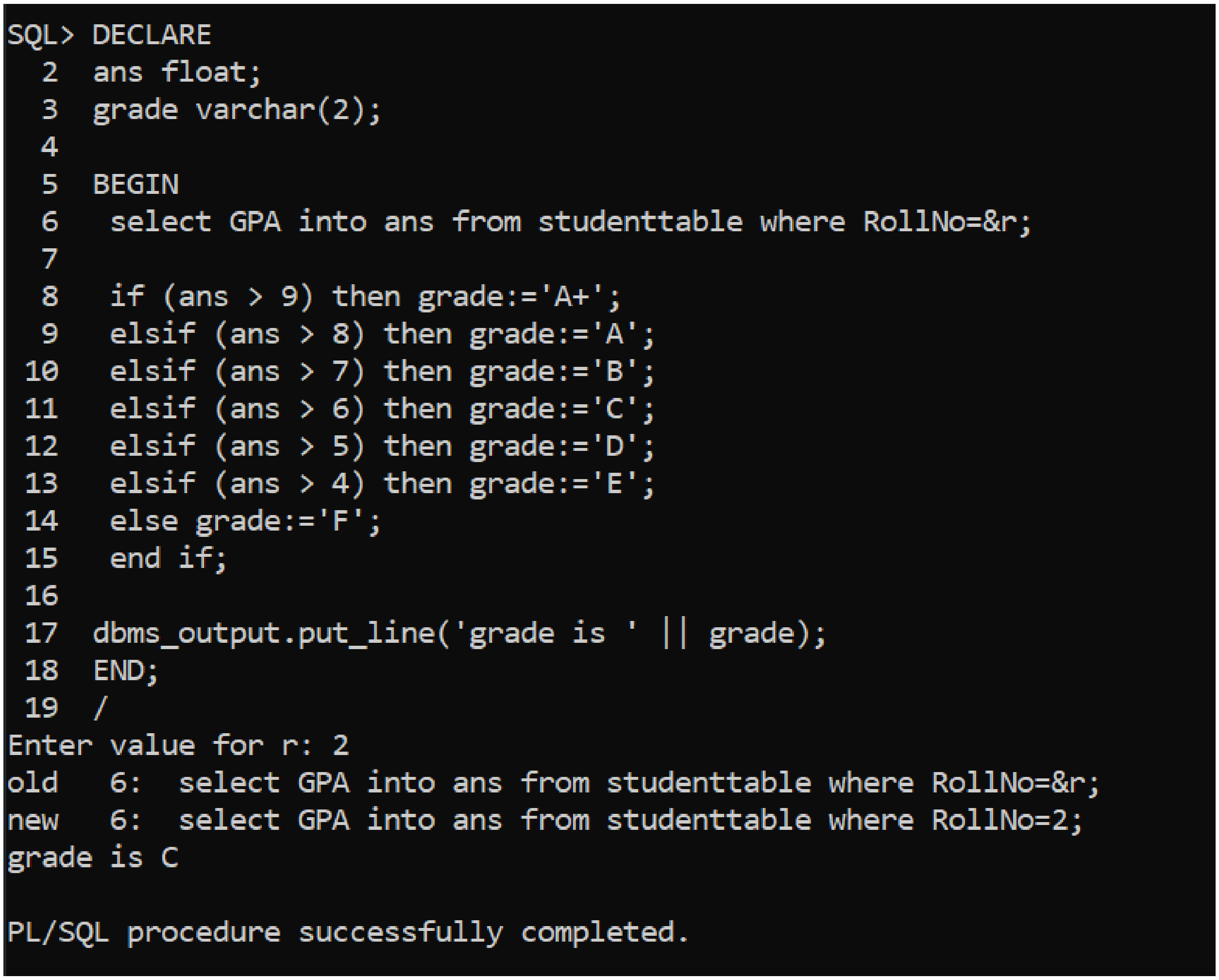
```
DECLARE
        ans float;
BEGIN

        select GPA into ans from studenttable where RollNo=&r;
        dbms_output.put_line('GPA is ' || ans);

END;
/
```



```
SQL> DECLARE
  2  ans float;
  3  BEGIN
  4
  5   select GPA into ans from studenttable where RollNo=&r;
  6  dbms_output.put_line('GPA is ' || ans);
  7
  8  END;
  9  /
Enter value for r: 2
old   5:  select GPA into ans from studenttable where RollNo=&r;
new   5:  select GPA into ans from studenttable where RollNo=2;
GPA is 6.5

PL/SQL procedure successfully completed.
```

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.

```
DECLARE
        ans float;
        grade varchar(2);

BEGIN
        select GPA into ans from studenttable where RollNo=&r;
        if (ans > 9) then grade:='A+';
        elsif (ans > 8) then grade:='A';
        elsif (ans > 7) then grade:='B';
        elsif (ans > 6) then grade:='C';
        elsif (ans > 5) then grade:='D';
        elsif (ans > 4) then grade:='E';
        else grade:='F';
        end if;

        dbms_output.put_line('grade is ' || grade);
END;
/
```



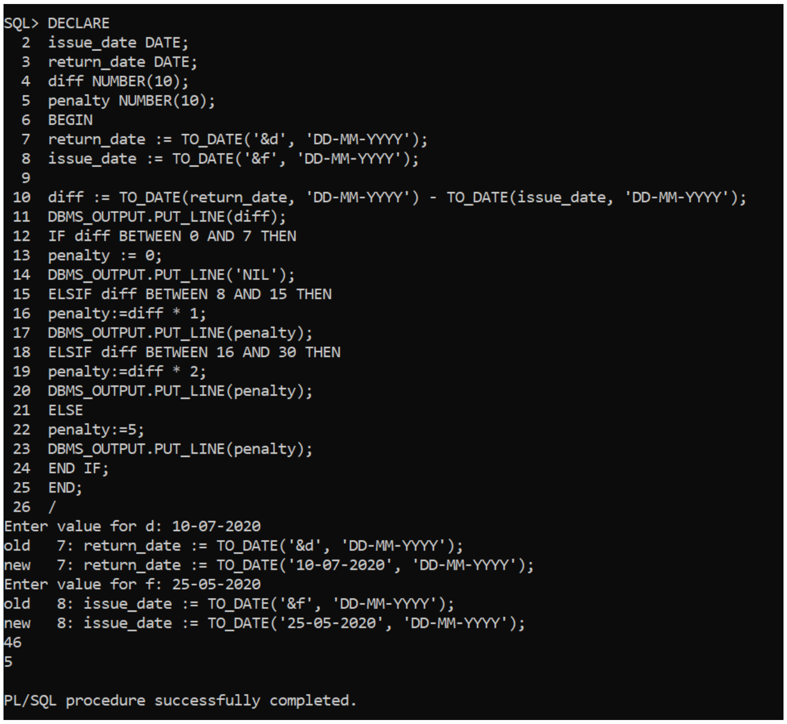
```
SQL> DECLARE
  2  ans float;
  3  grade varchar(2);
  4
  5  BEGIN
  6   select GPA into ans from studenttable where RollNo=&r;
  7
  8   if (ans > 9) then grade:='A+';
  9   elsif (ans > 8) then grade:='A';
 10   elsif (ans > 7) then grade:='B';
 11   elsif (ans > 6) then grade:='C';
 12   elsif (ans > 5) then grade:='D';
 13   elsif (ans > 4) then grade:='E';
 14   else grade:='F';
 15   end if;
 16
 17  dbms_output.put_line('grade is ' || grade);
 18  END;
 19  /
Enter value for r: 2
old   6:  select GPA into ans from studenttable where RollNo=&r;
new   6:  select GPA into ans from studenttable where RollNo=2;
grade is C

PL/SQL procedure successfully completed.
```

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

```
DECLARE
        issue_date DATE;
        return_date DATE;
        diff NUMBER(10);
        penalty NUMBER(10);
BEGIN
        return_date := TO_DATE('&d', 'DD-MM-YYYY');
        issue_date := TO_DATE('&f', 'DD-MM-YYYY');

        diff := TO_DATE(return_date, 'DD-MM-YYYY') - TO_DATE(issue_date, 'DD-MM-YYYY');
                DBMS_OUTPUT.PUT_LINE(diff);
        IF diff BETWEEN 0 AND 7 THEN
                penalty := 0;
                DBMS_OUTPUT.PUT_LINE('NIL');
        ELSIF diff BETWEEN 8 AND 15 THEN
                penalty:=diff * 1;
                DBMS_OUTPUT.PUT_LINE(penalty);
        ELSIF diff BETWEEN 16 AND 30 THEN
                penalty:=diff * 2;
                DBMS_OUTPUT.PUT_LINE(penalty);
        ELSE
                penalty:=5;
                DBMS_OUTPUT.PUT_LINE(penalty);
        END IF;
END;
/
```

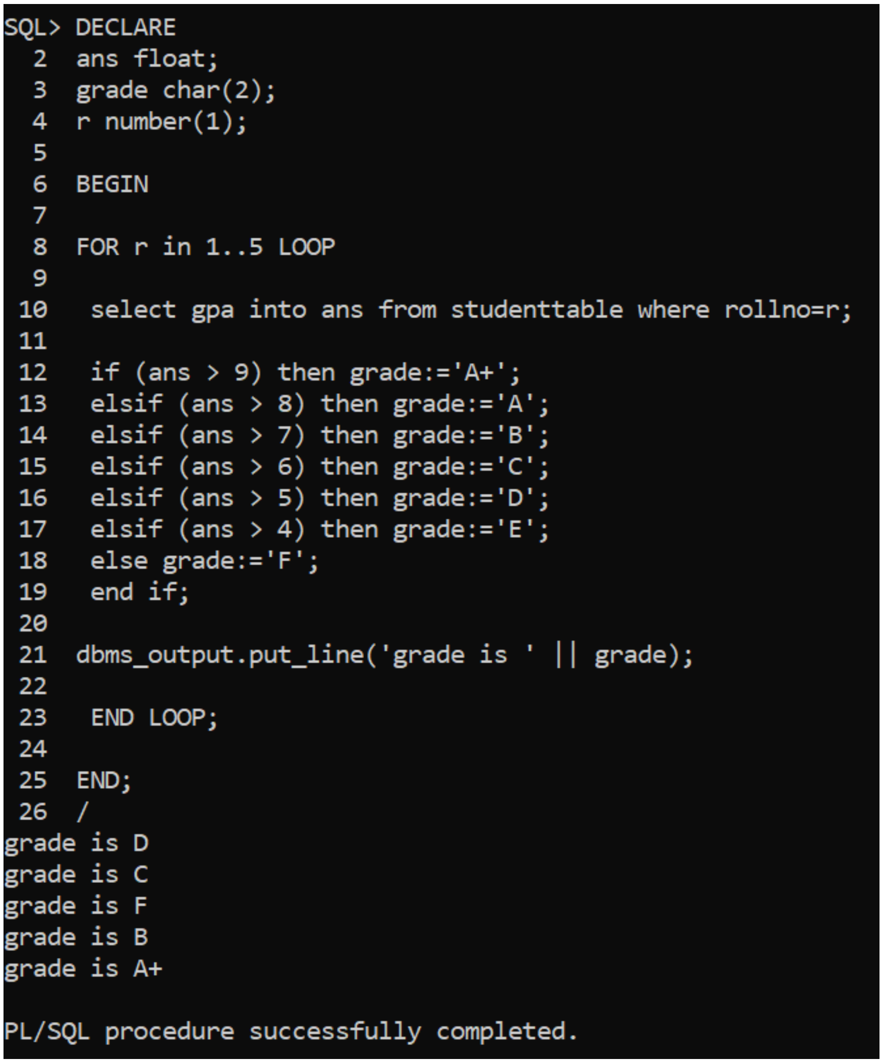


```
SQL> DECLARE
  2  issue_date DATE;
  3  return_date DATE;
  4  diff NUMBER(10);
  5  penalty NUMBER(10);
  6  BEGIN
  7  return_date := TO_DATE('&d', 'DD-MM-YYYY');
  8  issue_date := TO_DATE('&f', 'DD-MM-YYYY');
  9
 10  diff := TO_DATE(return_date, 'DD-MM-YYYY') - TO_DATE(issue_date, 'DD-MM-YYYY');
 11  DBMS_OUTPUT.PUT_LINE(diff);
 12  IF diff BETWEEN 0 AND 7 THEN
 13  penalty := 0;
 14  DBMS_OUTPUT.PUT_LINE('NIL');
 15  ELSIF diff BETWEEN 8 AND 15 THEN
 16  penalty:=diff * 1;
 17  DBMS_OUTPUT.PUT_LINE(penalty);
 18  ELSIF diff BETWEEN 16 AND 30 THEN
 19  penalty:=diff * 2;
 20  DBMS_OUTPUT.PUT_LINE(penalty);
 21  ELSE
 22  penalty:=5;
 23  DBMS_OUTPUT.PUT_LINE(penalty);
 24  END IF;
 25  END;
 26  /
Enter value for d: 10-07-2020
old   7: return_date := TO_DATE('&d', 'DD-MM-YYYY');
new   7: return_date := TO_DATE('10-07-2020', 'DD-MM-YYYY');
Enter value for f: 25-05-2020
old   8: issue_date := TO_DATE('&f', 'DD-MM-YYYY');
new   8: issue_date := TO_DATE('25-05-2020', 'DD-MM-YYYY');
46
5

PL/SQL procedure successfully completed.
```

Simple LOOP:
4. Write a PL/SQL block to print the letter grade of all the students(RollNo: 1 - 5).

```
DECLARE
        ans float;
        grade char(2);
        r number(1);

BEGIN

        FOR r in 1..5 LOOP

                select gpa into ans from studenttable where rollno=r;

                if (ans > 9) then grade:='A+';
                elsif (ans > 8) then grade:='A';
                elsif (ans > 7) then grade:='B';
                elsif (ans > 6) then grade:='C';
                elsif (ans > 5) then grade:='D';
                elsif (ans > 4) then grade:='E';
                else grade:='F';
                end if;

                dbms_output.put_line('grade is ' || grade);
```

```
    END LOOP;

END;
/
```



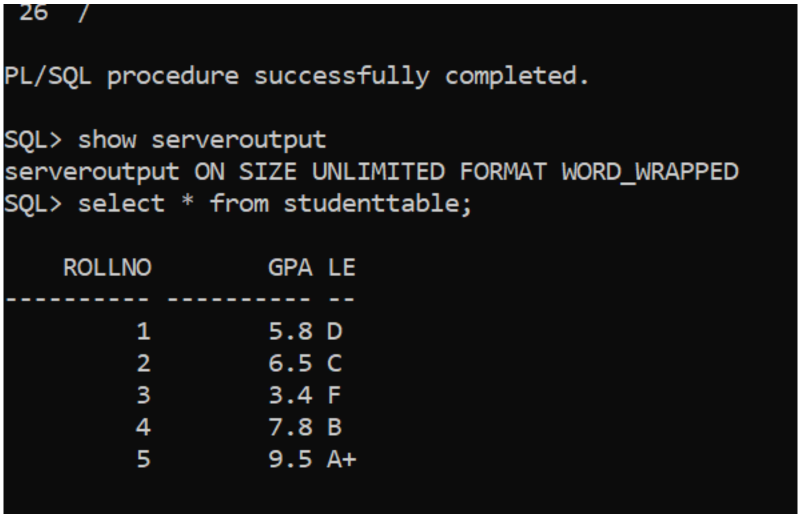
```
SQL> DECLARE
ans float;
grade char(2);
r number(1);

BEGIN

FOR r in 1..5 LOOP

 select gpa into ans from studenttable where rollno=r;

 if (ans > 9) then grade:='A+';
 elsif (ans > 8) then grade:='A';
 elsif (ans > 7) then grade:='B';
 elsif (ans > 6) then grade:='C';
 elsif (ans > 5) then grade:='D';
 elsif (ans > 4) then grade:='E';
 else grade:='F';
 end if;

dbms_output.put_line('grade is ' || grade);

 END LOOP;

END;
/
grade is D
grade is C
grade is F
grade is B
grade is A+

PL/SQL procedure successfully completed.
```

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.

```
DECLARE
    roll_number studenttable.rollno%TYPE;
    score studenttable.gpa%TYPE;
    n number(10);
    i number(10);
BEGIN
    roll_number:=1;
```

```
        i:=0;
        select count(rollno) into n  from studenttable;
        while i<n
        LOOP
        select gpa into score from studenttable where rollno = roll_number;
        IF score between 0 and 4  then
            update studenttable
                set LetterGrade = 'F' where rollno= roll_number;
        ELSIF score between 4 and 5 then
            update studenttable
                set LetterGrade = 'E' where rollno = roll_number;
        ELSIF score between 5 and 6 then
            update studenttable
                set LetterGrade = 'D' where rollno= roll_number;
        ELSIF score between 6 and 7 then
            update studenttable
                set LetterGrade = 'C' where rollno= roll_number;
        ELSIF score between 7 and 8 then
            update studenttable
                set LetterGrade = 'B' where rollno= roll_number;
        ELSIF score between 8 and 9 then
            update studenttable
                set LetterGrade = 'A' where rollno= roll_number;
        ELSE
            update studenttable
                set LetterGrade = 'A+' where rollno = roll_number;

        END IF;
        i:=i+1;
        roll_number:=roll_number+1;
        END LOOP;
    END;
```

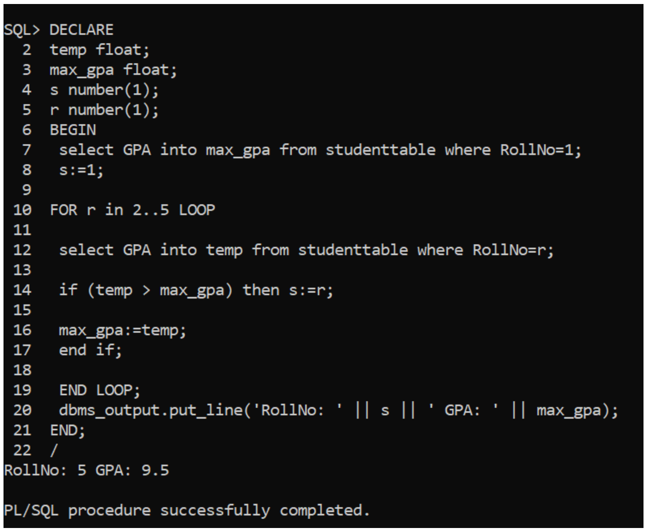


```
 26  /

PL/SQL procedure successfully completed.

SQL> show serveroutput
serveroutput ON SIZE UNLIMITED FORMAT WORD_WRAPPED
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+
```

Usage of FOR:

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

```
DECLARE
        temp float;
        max_gpa float;
        s number(1);
        r number(1);
BEGIN
        select GPA into max_gpa from studenttable where RollNo=1;
        s:=1;

        FOR r in 2..5 LOOP

                select GPA into temp from studenttable where RollNo=r;

                if (temp > max_gpa) then s:=r;

                max_gpa:=temp;
                end if;

        END LOOP;
        dbms_output.put_line('RollNo: ' || s || ' GPA: ' || max_gpa);
END;
/
```

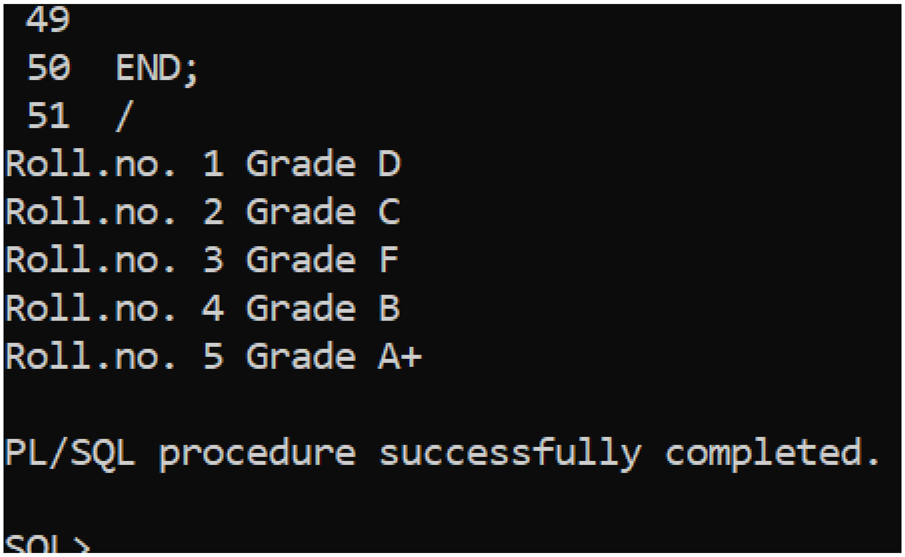


```
SQL> DECLARE
temp float;
max_gpa float;
s number(1);
r number(1);
BEGIN
 select GPA into max_gpa from studenttable where RollNo=1;
 s:=1;

FOR r in 2..5 LOOP

 select GPA into temp from studenttable where RollNo=r;

 if (temp > max_gpa) then s:=r;

 max_gpa:=temp;
 end if;

 END LOOP;
 dbms_output.put_line('RollNo: ' || s || ' GPA: ' || max_gpa);
END;
/
RollNo: 5 GPA: 9.5

PL/SQL procedure successfully completed.
```

Usage of GOTO:

7. Implement lab exercise 4 using GOTO.

```
DECLARE
        gp studenttable.gpa%TYPE;
        grade varchar(2);

BEGIN
        for i in 1..5 loop
                select gpa into gp from studenttable where rollno = i;

                if (gp>=9 and gp<=10) then goto ap;
                elsif (gp>=8 and gp<9) then goto aa;
                elsif (gp>=7 and gp<8) then goto bb;
                elsif (gp>=6 and gp<7) then goto cc;
                elsif (gp>=5 and gp<6) then goto dd;
                elsif (gp>=4 and gp<5) then goto ee;
                else goto ff;
                end if;

                <<ap>>
                        grade:='A+';
                        goto prnt;

                <<aa>>
                        grade:='A';
                        goto prnt;

                <<bb>>
                        grade:='B';
                        goto prnt;

                <<cc>>
                        grade:='C';
                        goto prnt;

                <<dd>>
                        grade:='D';
                        goto prnt;

                <<ee>>
                        grade:='E';
                        goto prnt;

                <<ff>>
                        grade:='F';

                <<prnt>>
                        dbms_output.put_line('Roll.no. ' || i || ' Grade ' || grade);

        end loop;

END;
/
```



```

 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>
```

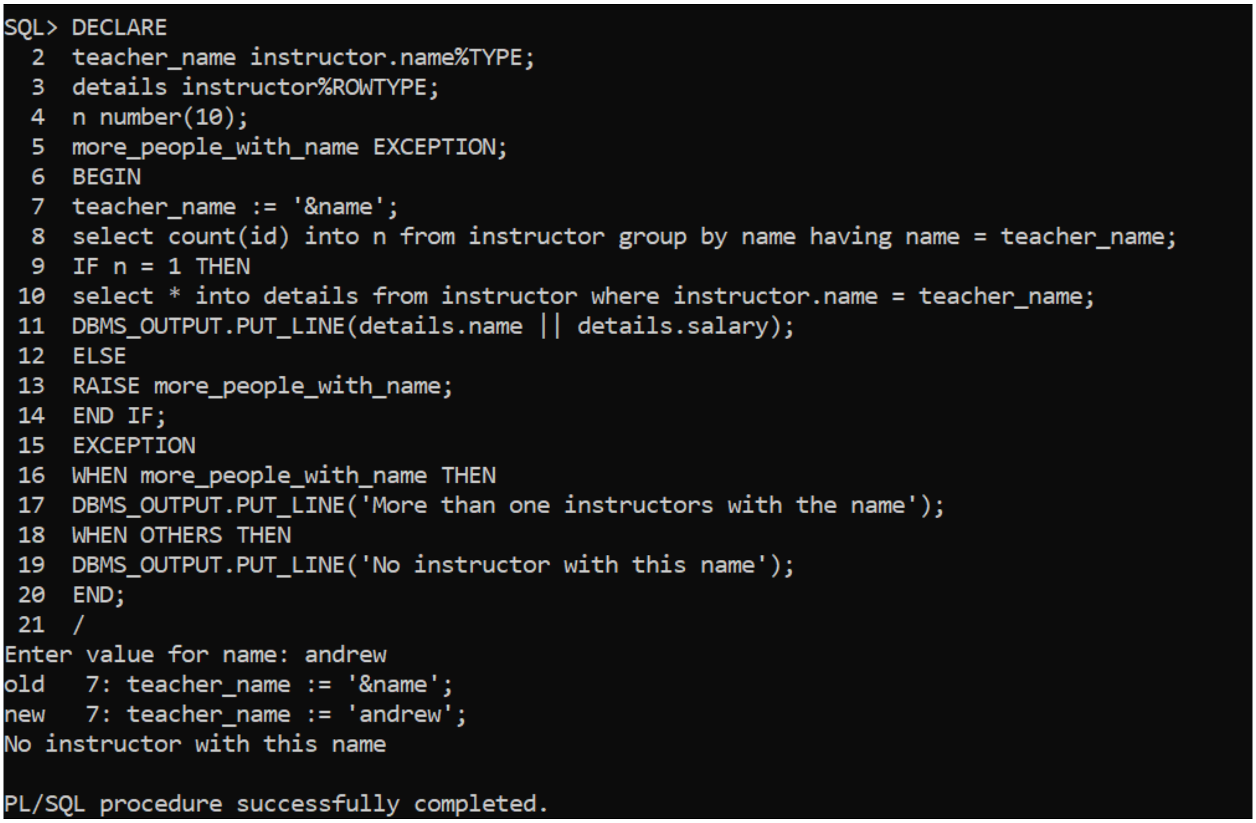
Exception Handling:

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:

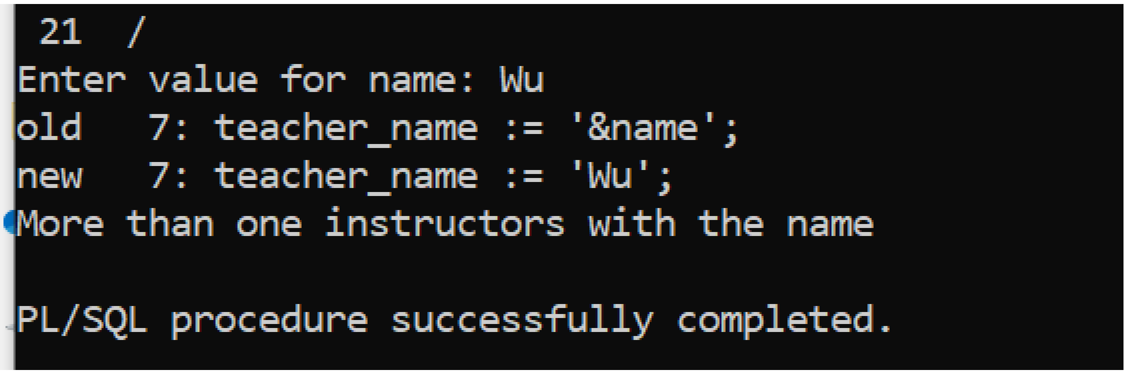
a. Multiple instructors with the same name

b. No instructor for the given name

```
DECLARE
        teacher_name instructor.name%TYPE;
        details instructor%ROWTYPE;
        n number(10);
        more_people_with_name EXCEPTION;
BEGIN
        teacher_name := '&name';
        select count(id) into n from instructor group by name having name = teacher_name;
        IF n = 1 THEN
                select * into details from instructor where instructor.name = teacher_name;
                DBMS_OUTPUT.PUT_LINE(details.name || details.salary);
        ELSE
                RAISE more_people_with_name;
        END IF;
EXCEPTION
        WHEN more_people_with_name THEN
                DBMS_OUTPUT.PUT_LINE('More than one instructors with the name');
        WHEN OTHERS THEN
                DBMS_OUTPUT.PUT_LINE('No instructor with this name');
END;
```



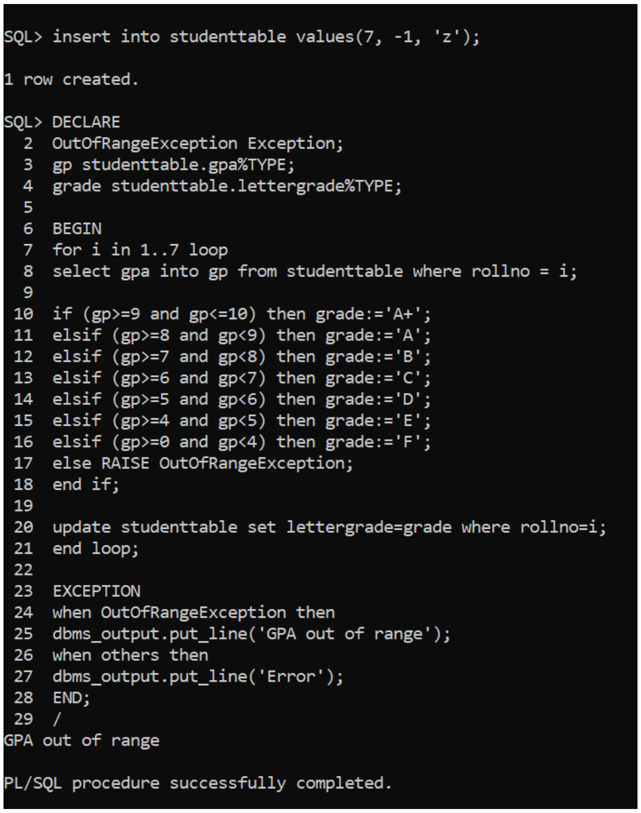
```
SQL> DECLARE
  2  teacher_name instructor.name%TYPE;
  3  details instructor%ROWTYPE;
  4  n number(10);
  5  more_people_with_name EXCEPTION;
  6  BEGIN
  7  teacher_name := '&name';
  8  select count(id) into n from instructor group by name having name = teacher_name;
  9  IF n = 1 THEN
 10  select * into details from instructor where instructor.name = teacher_name;
 11  DBMS_OUTPUT.PUT_LINE(details.name || details.salary);
 12  ELSE
 13  RAISE more_people_with_name;
 14  END IF;
 15  EXCEPTION
 16  WHEN more_people_with_name THEN
 17  DBMS_OUTPUT.PUT_LINE('More than one instructors with the name');
 18  WHEN OTHERS THEN
 19  DBMS_OUTPUT.PUT_LINE('No instructor with this name');
 20  END;
 21  /
Enter value for name: andrew
old   7: teacher_name := '&name';
new   7: teacher_name := 'andrew';
No instructor with this name

PL/SQL procedure successfully completed.
```



```
 21  /
Enter value for name: Wu
old   7: teacher_name := '&name';
new   7: teacher_name := 'Wu';
More than one instructors with the name

PL/SQL procedure successfully completed.
```

9. Extend lab exercise 5 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.

```
DECLARE
        OutOfRangeException Exception;
        gp studenttable.gpa%TYPE;
        grade studenttable.lettergrade%TYPE;

BEGIN
        for i in 1..7 loop
                select gpa into gp from studenttable where rollno = i;

                if (gp>=9 and gp<=10) then grade:='A+';
                elsif (gp>=8 and gp<9) then grade:='A';
                elsif (gp>=7 and gp<8) then grade:='B';
                elsif (gp>=6 and gp<7) then grade:='C';
                elsif (gp>=5 and gp<6) then grade:='D';
                elsif (gp>=4 and gp<5) then grade:='E';
                elsif (gp>=0 and gp<4) then grade:='F';
                else RAISE OutOfRangeException;
                end if;

                update studenttable set lettergrade=grade where rollno=i;
        end loop;

EXCEPTION
        when OutOfRangeException then
                dbms_output.put_line('GPA out of range');
        when others then
                dbms_output.put_line('Error');
END;
/
```



```
SQL> insert into studenttable values(7, -1, 'z');

1 row created.

SQL> DECLARE
OutOfRangeException Exception;
gp studenttable.gpa%TYPE;
grade studenttable.lettergrade%TYPE;

BEGIN
for i in 1..7 loop
select gpa into gp from studenttable where rollno = i;

if (gp>=9 and gp<=10) then grade:='A+';
elsif (gp>=8 and gp<9) then grade:='A';
elsif (gp>=7 and gp<8) then grade:='B';
elsif (gp>=6 and gp<7) then grade:='C';
elsif (gp>=5 and gp<6) then grade:='D';
elsif (gp>=4 and gp<5) then grade:='E';
elsif (gp>=0 and gp<4) then grade:='F';
else RAISE OutOfRangeException;
end if;

update studenttable set lettergrade=grade where rollno=i;
end loop;

EXCEPTION
when OutOfRangeException then
dbms_output.put_line('GPA out of range');
when others then
dbms_output.put_line('Error');
END;
/
GPA out of range

PL/SQL procedure successfully completed.
```