

WORKSHEET 2.2

Student Name: GAURAV KUMAR

UID: 21MCA3201

Branch: MCA

Section/Group: 9A

Semester: st Semester
1 2022

Date of Performance: 05/01/

Subject Name: PL/SQL LAB

Subject Code: 21CAP-607

1) Task to be _____ done:

WAP that creates a row-level trigger for the customers table that would fire for INSERT or DELETE operations performed on the CUSTOMERS table.

2) Steps for experiment/ _____ practical:

The screenshot shows a SQL IDE interface. At the top, there's a toolbar with 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command' and 'Find Tables'. Below the toolbar, a query is entered: `1 select * from customers1`. The 'Results' tab is active, displaying a table with 7 rows and 5 columns: ID, NAME, AGE, CITY, and SALARY. The data is as follows:

ID	NAME	AGE	CITY	SALARY
4	Gagan	20	Kolkata	31000
1	Aman	22	Mohali	25000
2	Akash	23	Mumbai	27000
3	Deepak	23	Kanpur	29000
5	Gopal	24	Delhi	30000
6	Shivam	26	Bangalore	33000
7	Ram	19	Ranchi	35000

At the bottom of the results table, it says '7 rows returned in 0.02 seconds' and there is a 'Download' button.

CREATING TRIGGER _____

```
CREATE OR REPLACE TRIGGER display_salary_change
BEFORE DELETE OR INSERT OR UPDATE ON customers1
FOR EACH ROW
WHEN (NEW.ID > 0)
DECLARE
```

```
sal_diff number;
```

```
BEGIN
```

```
sal_diff:= :NEW.salary - :OLD.salary;
```

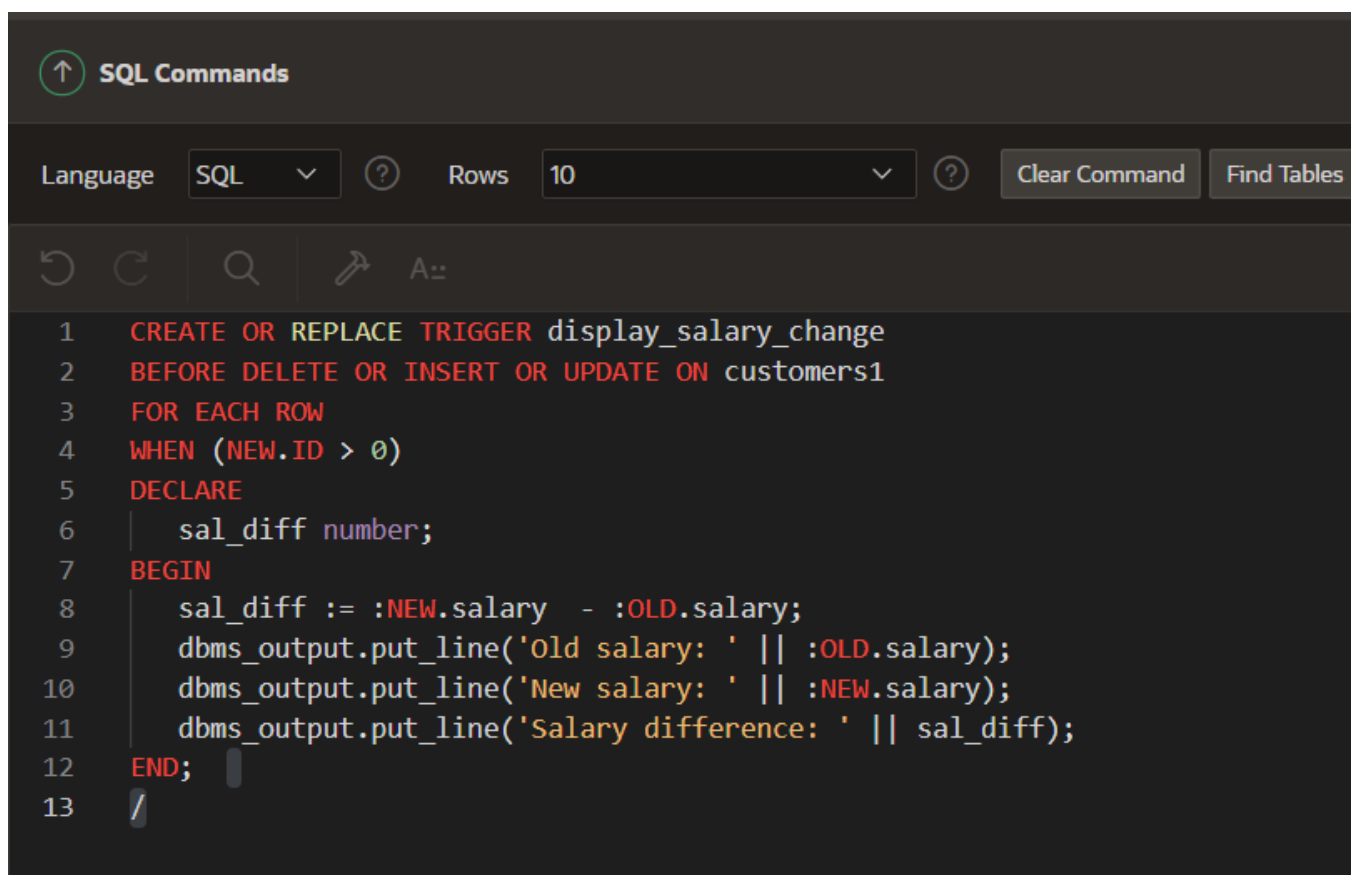
```
dbms_output.put_line('Old salary: ' || :OLD.salary);
```

```
dbms_output.put_line('New salary: ' || :NEW.salary);
```

```
dbms_output.put_line('Salary difference: ' || sal_diff);
```

```
END;
```

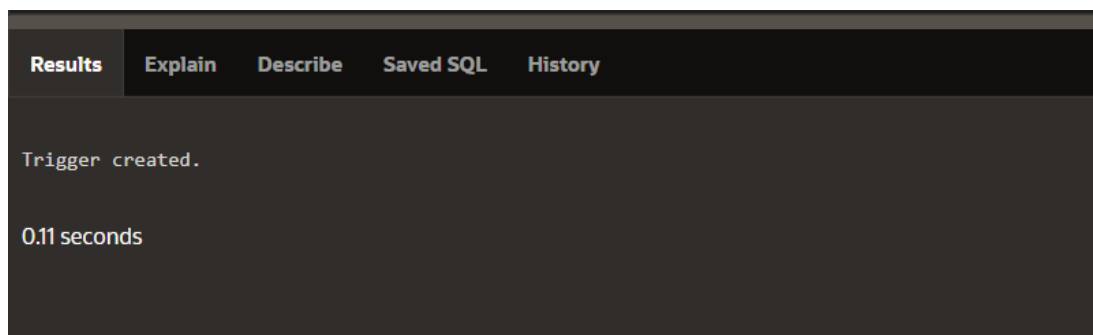
```
/
```



The screenshot shows the SQL Developer interface. At the top, there's a tab labeled 'SQL Commands'. Below it, the 'Language' is set to 'SQL' and 'Rows' is set to '10'. The main area displays the following PL/SQL code:

```
1 CREATE OR REPLACE TRIGGER display_salary_change
2 BEFORE DELETE OR INSERT OR UPDATE ON customers1
3 FOR EACH ROW
4 WHEN (NEW.ID > 0)
5 DECLARE
6     sal_diff number;
7 BEGIN
8     sal_diff := :NEW.salary - :OLD.salary;
9     dbms_output.put_line('Old salary: ' || :OLD.salary);
10    dbms_output.put_line('New salary: ' || :NEW.salary);
11    dbms_output.put_line('Salary difference: ' || sal_diff);
12 END;
13 /
```

OUTPUT



The screenshot shows the 'Results' window in SQL Developer. It has tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing the message 'Trigger created.' and the execution time '0.11 seconds'.

1) Triggering a trigger:-

DECLARE

total_rows number(

BEGIN

UPDATE customers1

SET salary = salary +

5000;
IF sql%notfound **THEN**

dbms_output.put_line('no customers

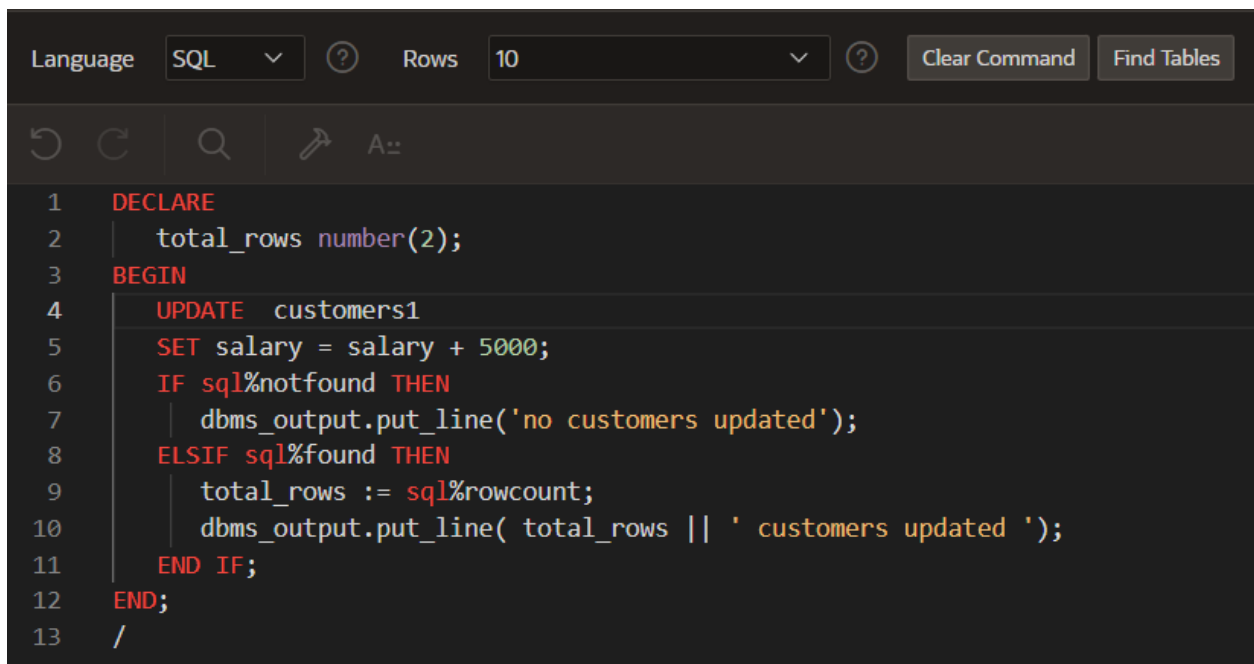
updated');
ELSIF sql%found **THEN**

total_rows := sql%

rowcount;
dbms_output.put_line(total_rows || ' customers

updated ');
END IF;

END;



```
Language SQL ? Rows 10 ? Clear Command Find Tables
1 DECLARE
2     total_rows number(2);
3 BEGIN
4     UPDATE customers1
5     SET salary = salary + 5000;
6     IF sql%notfound THEN
7         dbms_output.put_line('no customers updated');
8     ELSIF sql%found THEN
9         total_rows := sql%rowcount;
10        dbms_output.put_line( total_rows || ' customers updated ');
11    END IF;
12 END;
13 /
```

2) Output:

```

Results Explain Describe Saved SQL History

Old salary: 31000
New salary: 36000
Salary difference: 5000
Old salary: 25000
New salary: 30000
Salary difference: 5000
Old salary: 27000
New salary: 32000
Salary difference: 5000
Old salary: 29000
New salary: 34000
Salary difference: 5000
Old salary: 30000
New salary: 35000
Salary difference: 5000
Old salary: 33000
New salary: 38000
Salary difference: 5000
Old salary: 35000
New salary: 40000
Salary difference: 5000
7 customers updated

1 row(s) updated.

```

4) Learning outcomes (What I have learnt):

1. Implementation of Trigger
2. Triggering a Trigger.

Evaluation Grid:

Sr. No.	Parameters	Marks	Maximum
1.	Demonstration and Performance	Obtained	Marks
2.	Worksheet		10
3.	Post Lab Quiz		5