EXPERIMENT-15

AIM : TO Write PL/SQL program to implement Trigger on table.

Trigger is invoked by Oracle engine automatically whenever a specified event occurs. Trigger is

stored into database and invoked repeatedly, when specific condition match. Triggers are

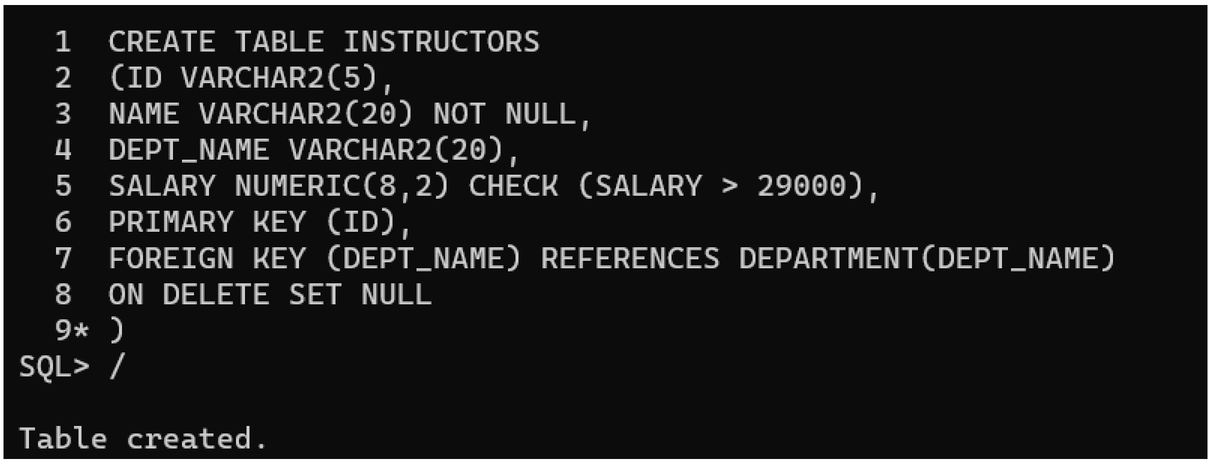
stored programs, which are automatically executed or fired when some event occurs. Triggers

are written to be executed in response to any of the following events.

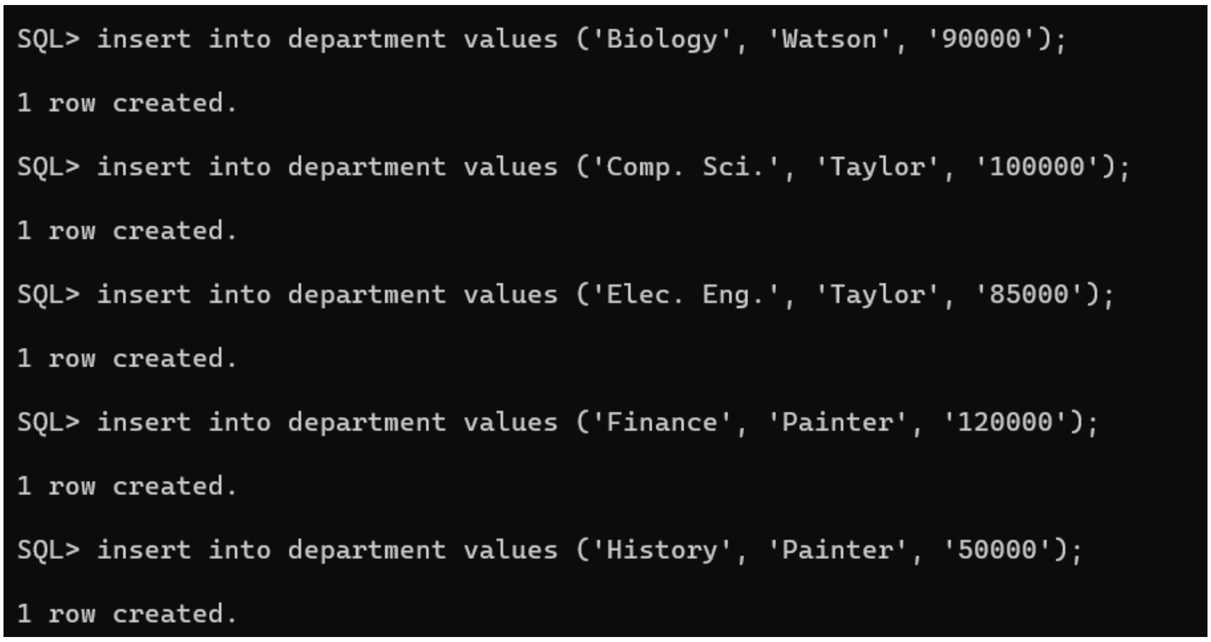
A database manipulation (DML) statement (DELETE, INSERT, or UPDATE).

A database definition (DDL) statement (CREATE, ALTER, or DROP).

A database operation (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

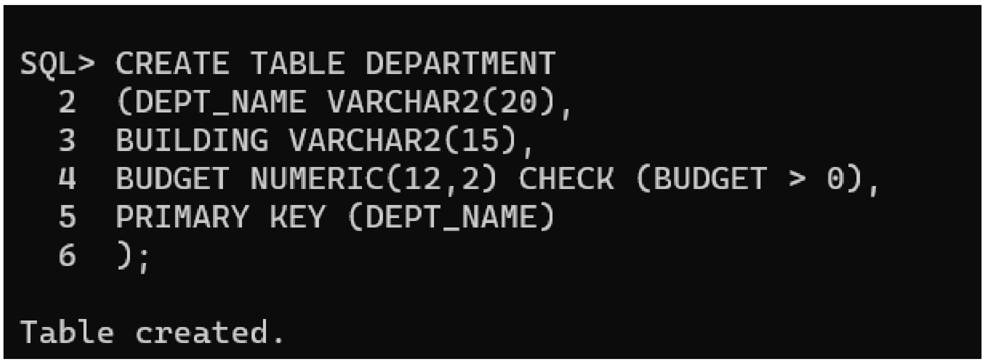


```
CREATE TABLE INSTRUCTORS
(ID VARCHAR2(5),
NAME VARCHAR2(20) NOT NULL,
DEPT_NAME VARCHAR2(20),
SALARY NUMERIC(8,2) CHECK (SALARY > 29000),
PRIMARY KEY (ID),
FOREIGN KEY (DEPT_NAME) REFERENCES DEPARTMENT(DEPT_NAME)
ON DELETE SET NULL
)
SQL> /

Table created.
```



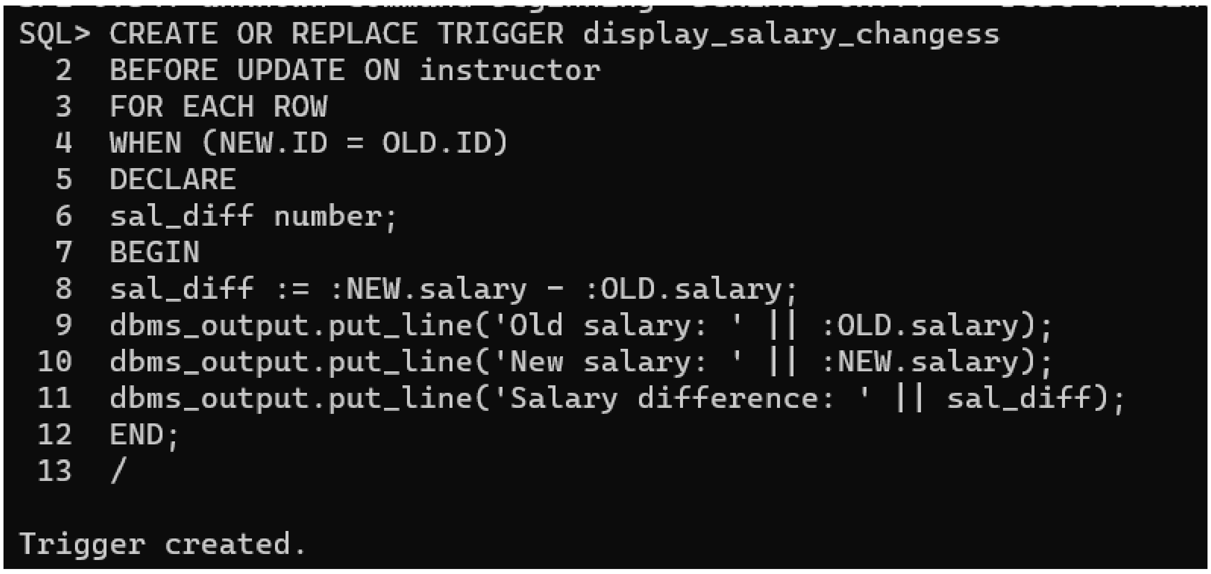
```
SQL> insert into department values ('Biology', 'Watson', '90000');

1 row created.

SQL> insert into department values ('Comp. Sci.', 'Taylor', '100000');

1 row created.

SQL> insert into department values ('Elec. Eng.', 'Taylor', '85000');

1 row created.

SQL> insert into department values ('Finance', 'Painter', '120000');

1 row created.

SQL> insert into department values ('History', 'Painter', '50000');

1 row created.
```

CREATING DEPARTMENT TABLE :



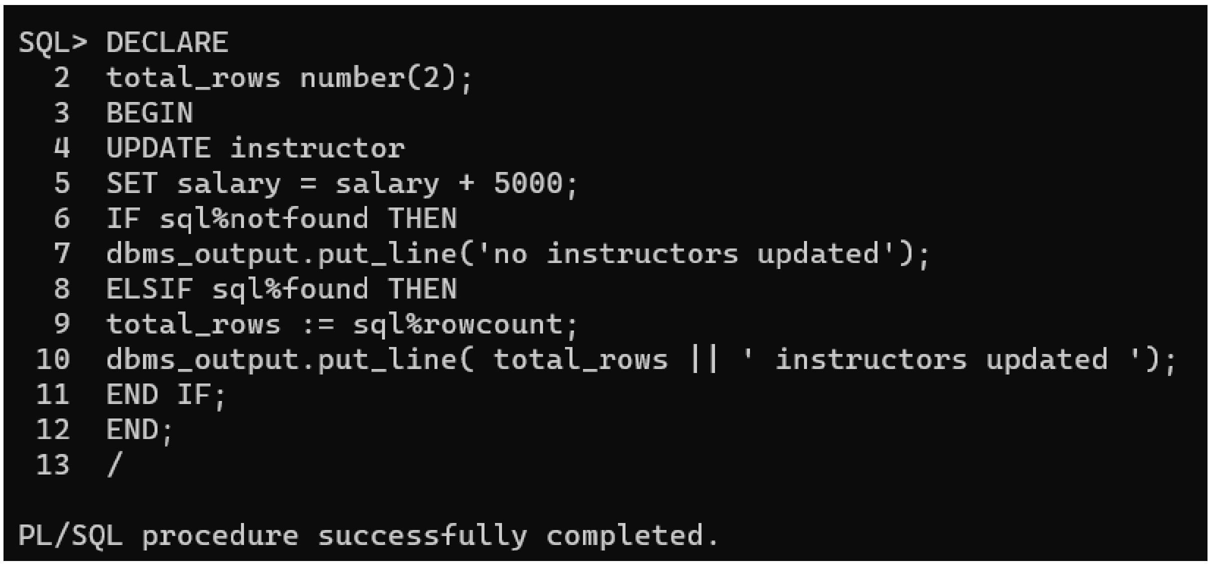
```
SQL> CREATE TABLE DEPARTMENT
(DEPT_NAME VARCHAR2(20),
BUILDING VARCHAR2(15),
BUDGET NUMERIC(12,2) CHECK (BUDGET > 0),
PRIMARY KEY (DEPT_NAME)
);

Table created.
```

An example to create Trigger :



```
SQL> CREATE OR REPLACE TRIGGER display_salary_changess
BEFORE UPDATE ON instructor
FOR EACH ROW
WHEN (NEW.ID = OLD.ID)
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;
/

Trigger created.
```

A PL/SQL Procedure to execute a trigger:



```
SQL> DECLARE
     total_rows number(2);
     BEGIN
     UPDATE instructor
     SET salary = salary + 5000;
     IF sql%notfound THEN
     dbms_output.put_line('no instructors updated');
     ELSIF sql%found THEN
     total_rows := sql%rowcount;
     dbms_output.put_line( total_rows || ' instructors updated ');
     END IF;
     END;
     /

PL/SQL procedure successfully completed.
```

Conclusion:

The pl/sql program is successfully executed.