Name : Siva Rama Krishna Nallapati
ID NO : 190031154

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Lab-4 : Triggers in Database Security-2

Pre-Lab:

Q1. What is auditing in Oracle?

Sol) Auditing is the monitoring and recording of selected user database actions. It can be based on individual actions, such as the type of SQL statement executed, or on combinations of factors that can include user name, application, time, and so on. Security policies can trigger auditing when specified elements in an Oracle database are accessed or altered, including the contents within a specified object.

Q2. What are the different types of auditing present in Oracle?

Sol) Statement Auditing, Privilege Auditing, Schema Object Auditing, Fine-Grained Auditing.

Q3. What is the importance of auditing?

Sol) Auditing is a means of evaluating the effectiveness of a company's internal controls. Maintaining an effective system of internal controls is vital for achieving a company's business objectives, obtaining reliable financial reporting on its operations, preventing fraud and misappropriation of its assets, and minimizing its cost of capital.

Name : Siva Rama Krishna Nallapati
ID NO : 190031154

In-Lab:

Q1. Create a table ‘emp_data_labtrig2’ with the column ‘eno’, ‘ename’, ‘job’, ‘hire_day’ and ‘salary’ and insert the following data:

‘eno’ is the primary key.

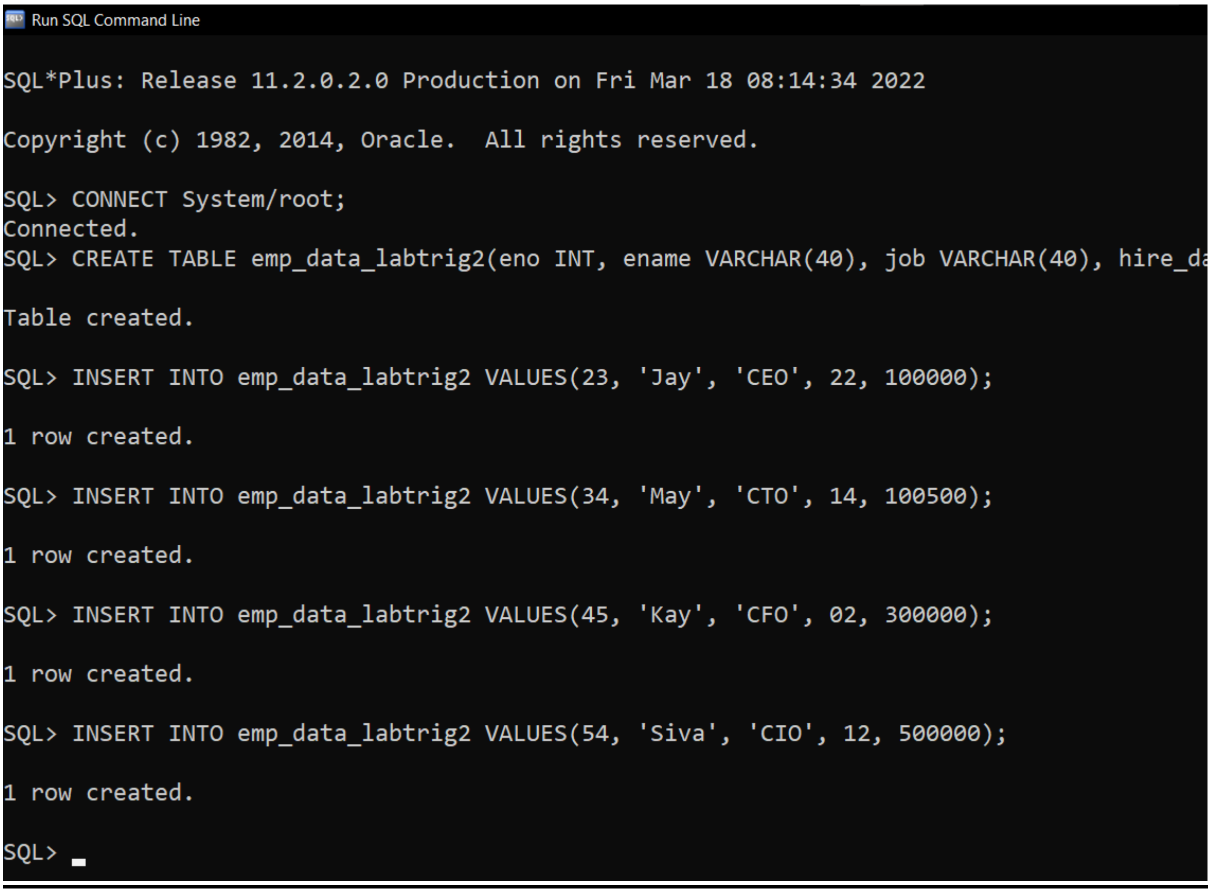
ENO	ENAME	JOB	HIRE_DAY	SALARY
23	Jay	CEO	22	100000
34	May	CTO	14	100500
45	Kay	CFO	02	300000

CREATE TABLE emp_data_labtrig2 (ENO INT, ENAME VARCHAR(40), JOB VARCHAR(40), HIRE_DAY NUMBER, SALARY INT, PRIMARY KEY(ENO));

INSERT INTO emp_data_labtrig2 VALUES(23, 'JAY', 'CEO', 22, 100000);

INSERT INTO emp_data_labtrig2 VALUES(34, 'MAY', 'CTO', 14, 100500);

INSERT INTO emp_data_labtrig2 VALUES(45, 'KAY', 'CFO', 02, 300000);

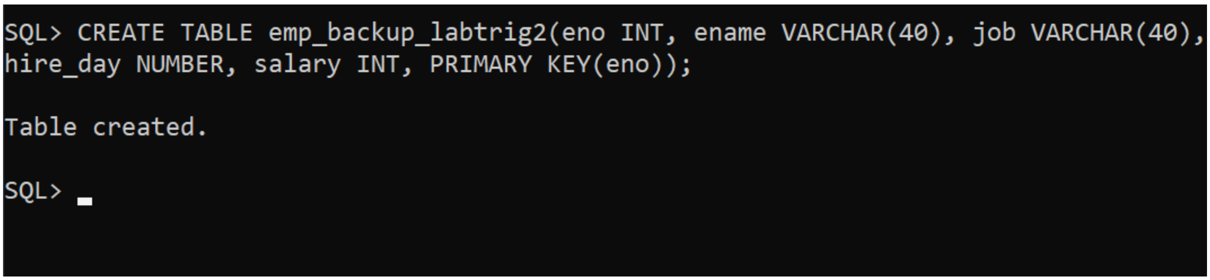


```
SQL*Plus: Release 11.2.0.2.0 Production on Fri Mar 18 08:14:34 2022

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

SQL> CONNECT System/root;
Connected.
SQL> CREATE TABLE emp_data_labtrig2(eno INT, ename VARCHAR(40), job VARCHAR(40), hire_da

Table created.

SQL> INSERT INTO emp_data_labtrig2 VALUES(23, 'Jay', 'CEO', 22, 100000);

1 row created.

SQL> INSERT INTO emp_data_labtrig2 VALUES(34, 'May', 'CTO', 14, 100500);

1 row created.

SQL> INSERT INTO emp_data_labtrig2 VALUES(45, 'Kay', 'CFO', 02, 300000);

1 row created.

SQL> INSERT INTO emp_data_labtrig2 VALUES(54, 'Siva', 'CIO', 12, 500000);

1 row created.

SQL> _
```

Name : Siva Rama Krishna Nallapati
ID NO : 190031154

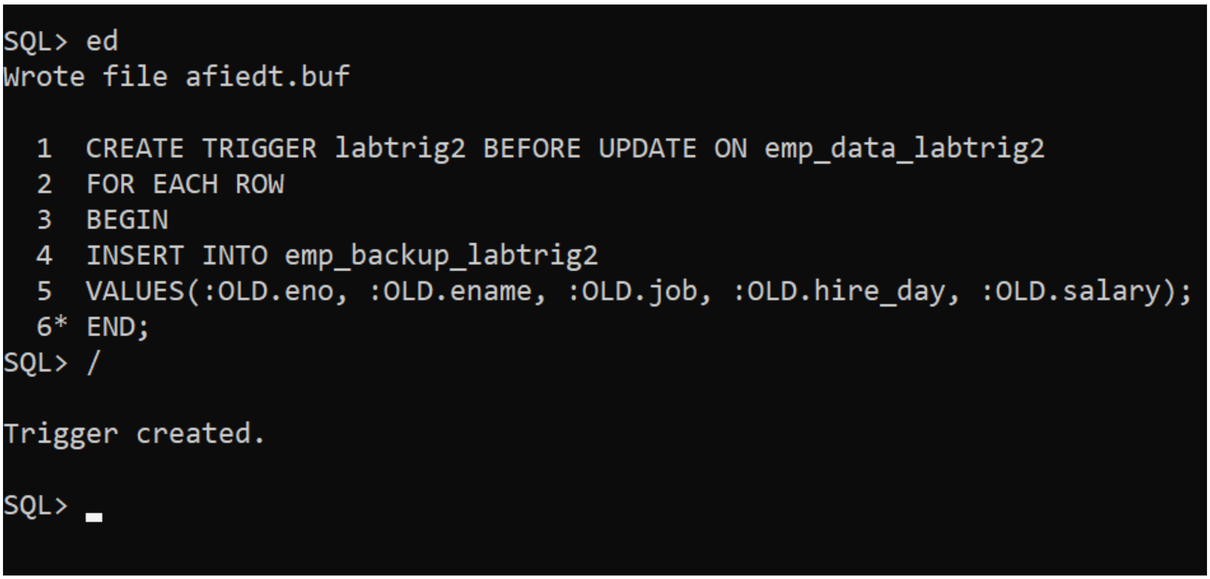
Q2. Create a table called ‘emp_backup_labtrig2’ with the same columns as ‘emp_data_labtrig2’. This will be the audit table. Then create a trigger ‘labtrig2’ which will work before updation in ‘emp_data_labtrig2’ table and create a copy of the record to be updated in the table ‘emp_backup_labtrig2’

Sol) Connect to system and create the table ‘emp_backup_labtrig2’. create table emp_backup_labtrig2 (eno int, ename varchar(40), job varchar(40), hire_day number, salary int, primary key(eno));



```
SQL> CREATE TABLE emp_backup_labtrig2(eno INT, ename VARCHAR(40), job VARCHAR(40),
hire_day NUMBER, salary INT, PRIMARY KEY(eno));

Table created.

SQL>
```

Type ‘ed’ in command line and create the trigger ‘labtrig2’. Then press ‘ALT+F+X’ and save when prompted. Then type ‘/’ (slash) in the command line.

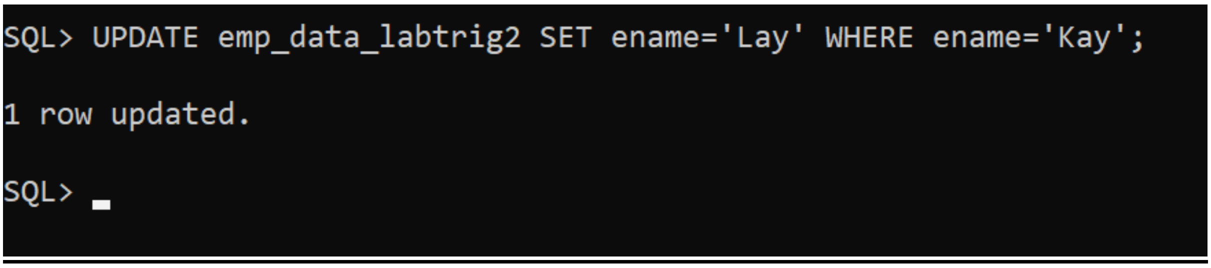


```
SQL> ed
Wrote file afiedt.buf

     CREATE TRIGGER labtrig2 BEFORE UPDATE ON emp_data_labtrig2
     FOR EACH ROW
     BEGIN
     INSERT INTO emp_backup_labtrig2
     VALUES(:OLD.eno, :OLD.ename, :OLD.job, :OLD.hire_day, :OLD.salary);
     END;
SQL> /

Trigger created.

SQL>
```

Q3. Fire the trigger ‘labtrig2’ and test whether it works by updating the row where the employee name is ‘Kay’ and check the ‘emp_backup_labtrig2’ table and write the output. (That means you have to update the required row and display emp_backup_labtrig2 to see if the updated row has been copied into the table successfully.)

Update the required row.

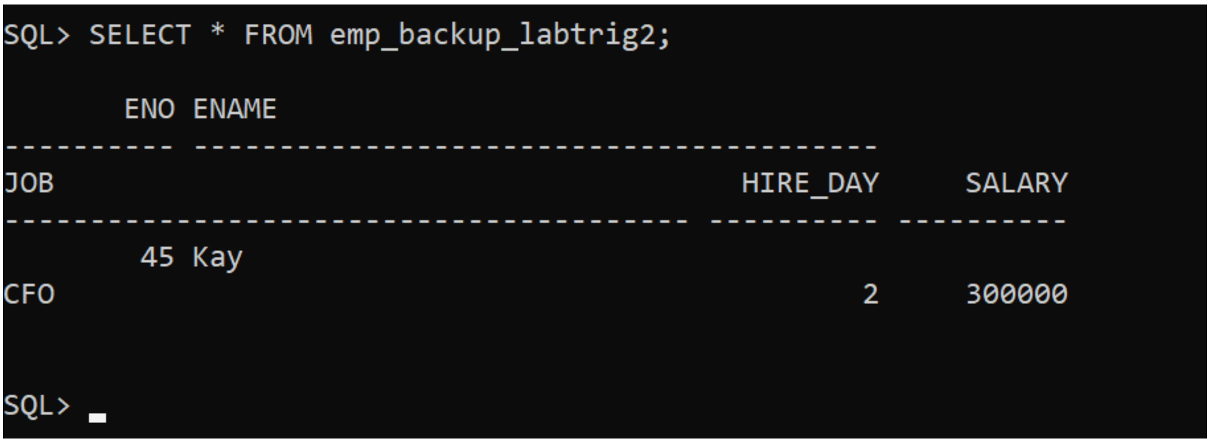
UPDATE emp_data_labtrig2 SET ENAME=’LAY’ WHERE ENAME='KAY';



```
SQL> UPDATE emp_data_labtrig2 SET ename='Lay' WHERE ename='Kay';

1 row updated.

SQL> _
```

Display the ‘emp_backup_labtrig2’ table.

SELECT * FROM emp_backup_labtrig2;



```
SQL> SELECT * FROM emp_backup_labtrig2;

       ENO ENAME
---------- ----------------------------------------
JOB                                      HIRE_DAY     SALARY
---------------------------------------- ---------- ----------
        45 Kay
CFO                                              2     300000


SQL> _
```

Q4. Create dept table and deptaudit table.

Dept table -

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Deptaudit table *structure* –

Name	Null?	Type
DEPTNO		NUMBER
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)

CHANGE_TYPE		VARCHAR2(1)
CHANGED_BY		VARCHAR2(30)
CHANGED_TIME		DATE

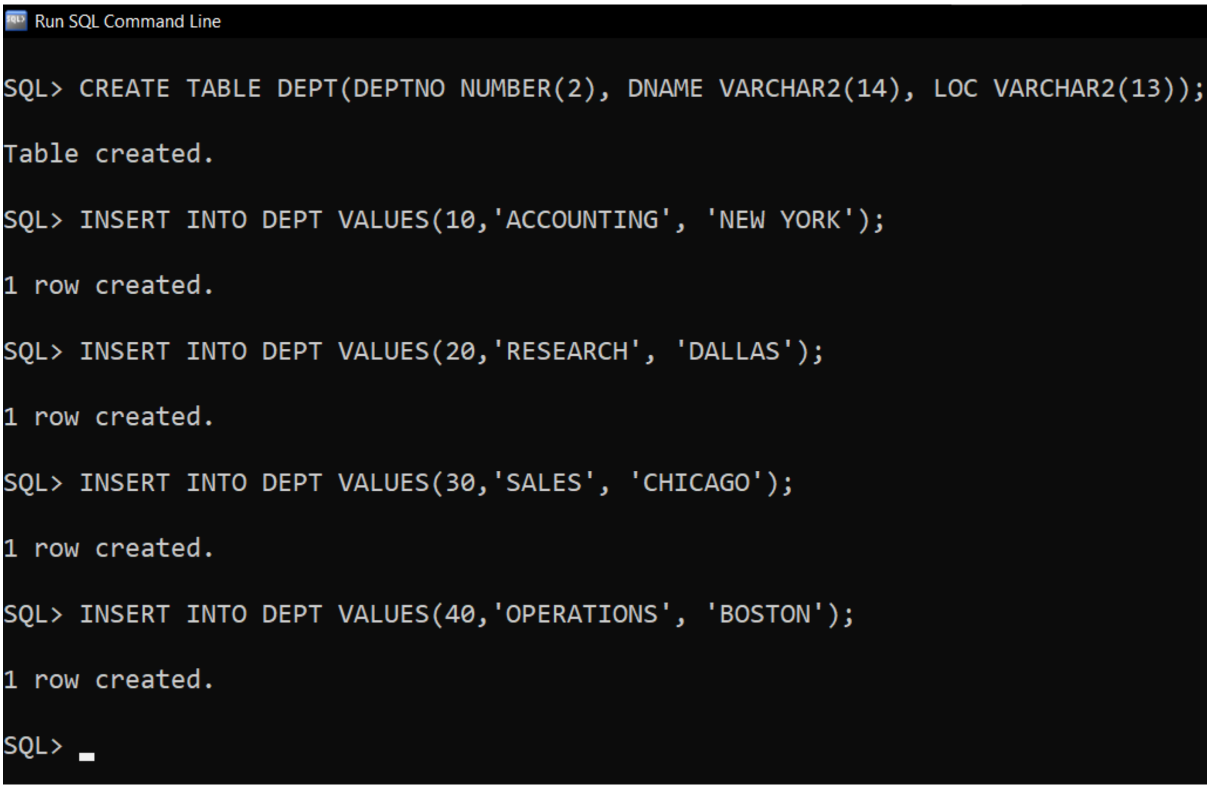
Sol)

Create the ‘dept’ table.

CREATE TABLE DEPT(DEPTNO NUMBER(2), DNAME VARCHAR2(14), LOC VARCHAR2(13));

2. Insert the data into ‘dept’ table.

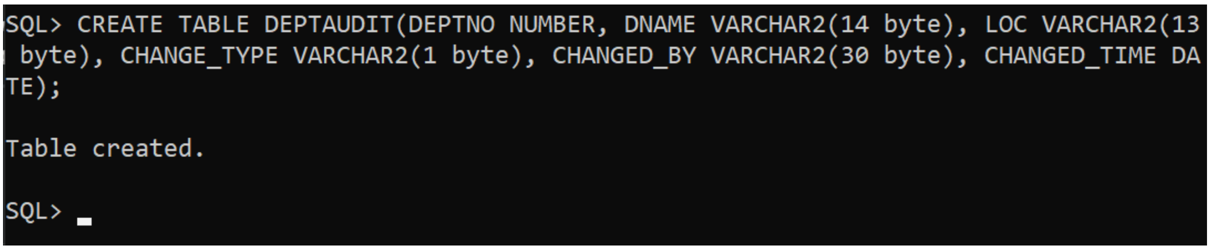
INSERT INTO DEPT VALUES (10, 'ACCOUNTING', 'NEW YORK'); INSERT INTO DEPT VALUES (20, 'RESEARCH', 'DALLAS'); INSERT INTO DEPT VALUES (30, 'SALES', 'CHICAGO'); INSERT INTO DEPT VALUES (40, 'OPERATIONS', 'BOSTON');



```
SQL> CREATE TABLE DEPT(DEPTNO NUMBER(2), DNAME VARCHAR2(14), LOC VARCHAR2(13));

Table created.

SQL> INSERT INTO DEPT VALUES(10,'ACCOUNTING', 'NEW YORK');

1 row created.

SQL> INSERT INTO DEPT VALUES(20,'RESEARCH', 'DALLAS');

1 row created.

SQL> INSERT INTO DEPT VALUES(30,'SALES', 'CHICAGO');

1 row created.

SQL> INSERT INTO DEPT VALUES(40,'OPERATIONS', 'BOSTON');

1 row created.

SQL>
```
]

3. Create the ‘deptaudit’ table.

CREATE TABLE DEPTAUDIT (DEPTNO NUMBER, DNAME VARCHAR2(14 byte), LOC VARCHAR2(13 byte), CHANGE_TYPE VARCHAR2(1 byte), CHANGED_BY VARCHAR2(30 byte), CHANGED_TIME DATE);



```
SQL> CREATE TABLE DEPTAUDIT(DEPTNO NUMBER, DNAME VARCHAR2(14 byte), LOC VARCHAR2(13
 byte), CHANGE_TYPE VARCHAR2(1 byte), CHANGED_BY VARCHAR2(30 byte), CHANGED_TIME DA
TE);

Table created.

SQL>
```

Q5. Create a trigger ‘auditDEPTAR’ that audits the operations performed on the ‘dept’ table.

The trigger should work after any insert, update or delete operation.

It must log those operations in a table called ‘deptaudit’. It should display ‘I’ for insert, ‘U’ for update and ‘D’ for delete.

It must also log the data (i.e. ‘deptno’, ‘dname’, ‘loc’) as well as who changed the data and when it was changed.

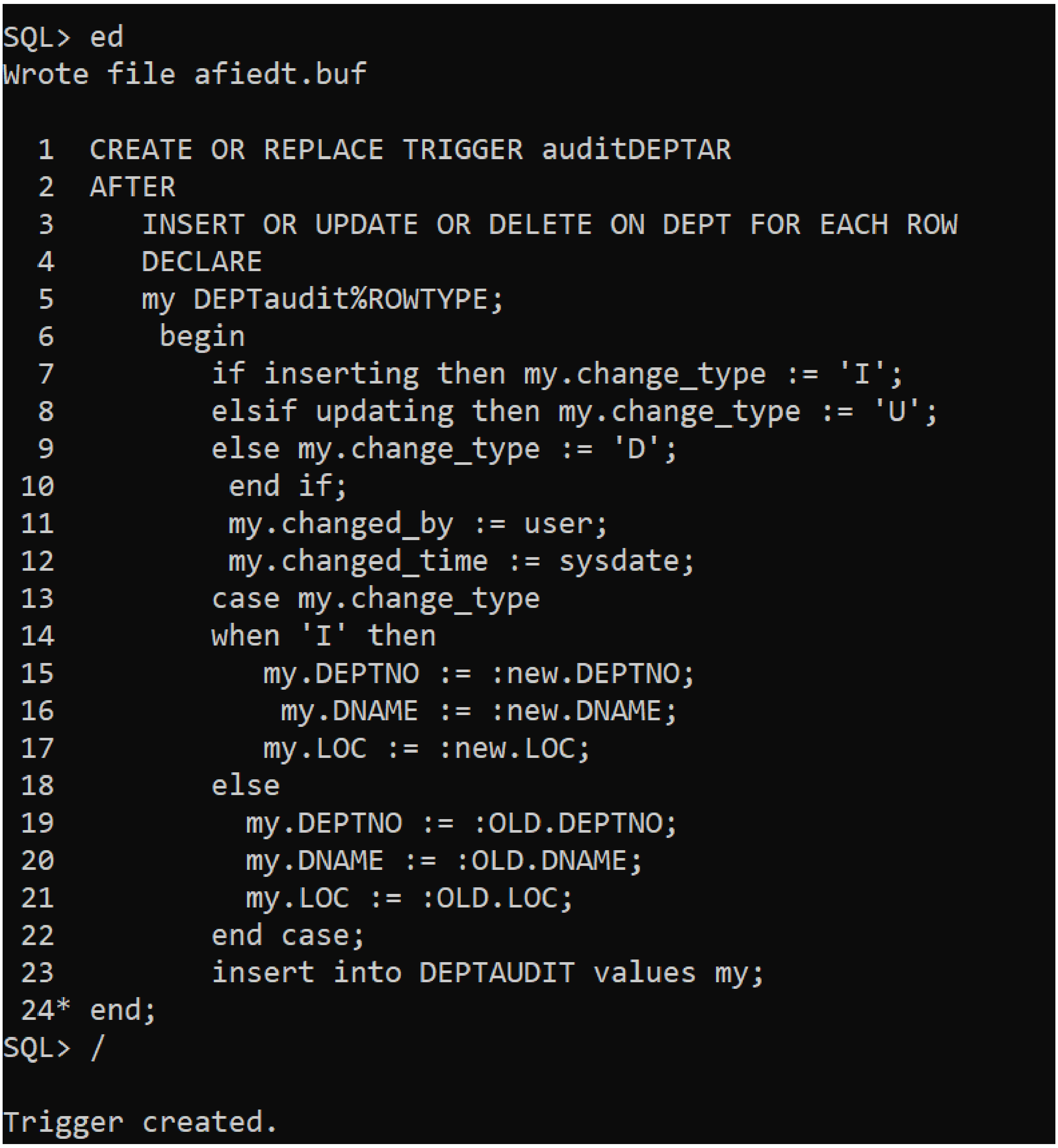
The output is stored in the table ‘deptaudit’.

Check by inserting the data 50, TEST, TESTS in ‘dept’ table.

Sample Output-

```
DEPTNO DNAME LOC      C CHANGED_BY           CHANGED_T
---------------------       ----------- --------------------            -----------------
 50       TEST    TESTS       I      SYSTEM                   29-DEC-2019
```

Name : Siva Rama Krishna Nallapati
ID NO : 190031154

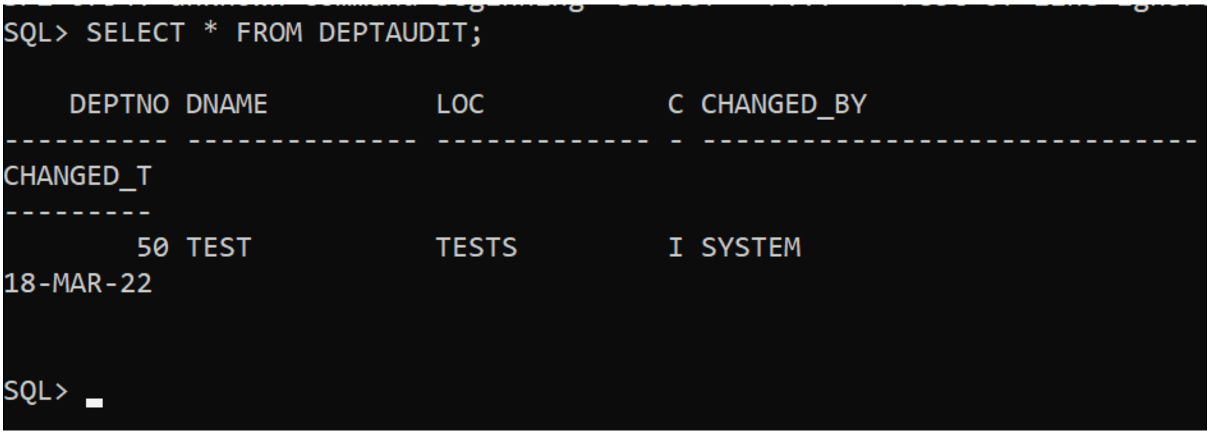


```
SQL> ed
Wrote file afiedt.buf

CREATE OR REPLACE TRIGGER auditDEPTAR
AFTER
   INSERT OR UPDATE OR DELETE ON DEPT FOR EACH ROW
   DECLARE
   my DEPTaudit%ROWTYPE;
    begin
       if inserting then my.change_type := 'I';
       elsif updating then my.change_type := 'U';
       else my.change_type := 'D';
        end if;
        my.changed_by := user;
        my.changed_time := sysdate;
       case my.change_type
       when 'I' then
          my.DEPTNO := :new.DEPTNO;
           my.DNAME := :new.DNAME;
          my.LOC := :new.LOC;
       else
         my.DEPTNO := :OLD.DEPTNO;
         my.DNAME := :OLD.DNAME;
         my.LOC := :OLD.LOC;
       end case;
       insert into DEPTAUDIT values my;
end;
SQL> /

Trigger created.
```

Name : Siva Rama Krishna Nallapati
ID NO : 190031154

Q6. Display the audit table after insertion of data and write the output.

Sol) SELECT * FROM DEPTAUDIT



```
SQL> SELECT * FROM DEPTAUDIT;

    DEPTNO DNAME          LOC           C CHANGED_BY
---------- -------------- ------------- - ------------------------------
CHANGED_T
---------
        50 TEST           TESTS         I SYSTEM
18-MAR-22


SQL> _
```

Name : Siva Rama Krishna Nallapati
ID NO : 190031154

Post-Lab

Q1. Create a table ‘bank_transactions’ with the column ‘TID’, ‘TNO’, ‘NAME’, ‘CREATOR’ and insert the following data:

‘TID’ is the primary key

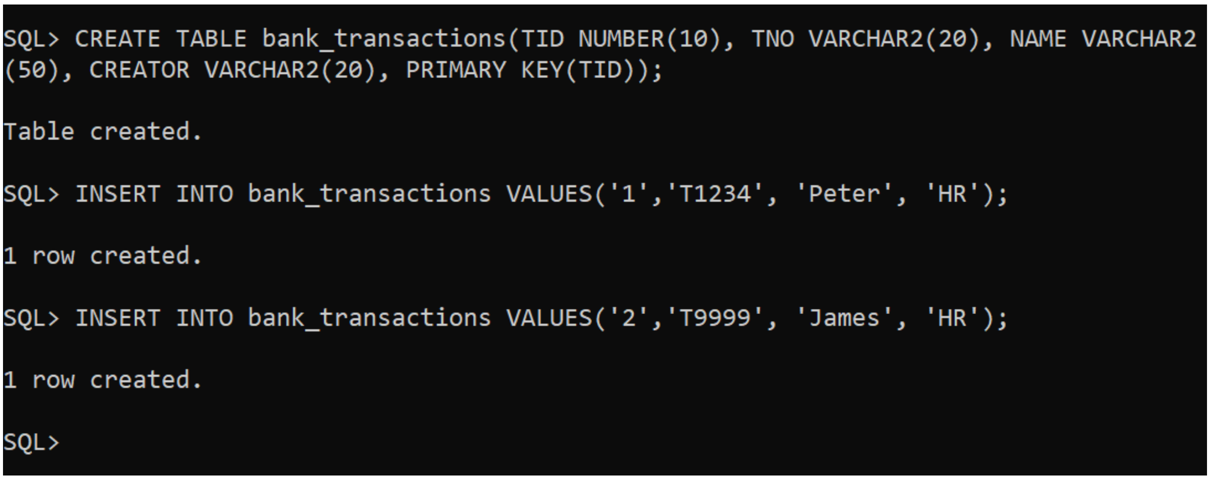
TID	TNO	NAME	CREATOR
1	T1234	Peter	HR
2	T9999	James	HR

Sol)

1. CREATE TABLE bank_transactions(TID number(10) primary key, TNO varchar2(20), NAME varchar2(50), CREATOR varchar2(20));

2. INSERT INTO BANK_TRANSACTIONS values ('1','T1234','Peter','HR');

   INSERT INTO BANK_TRANSACTIONS values ('2','T9999','James','HR');

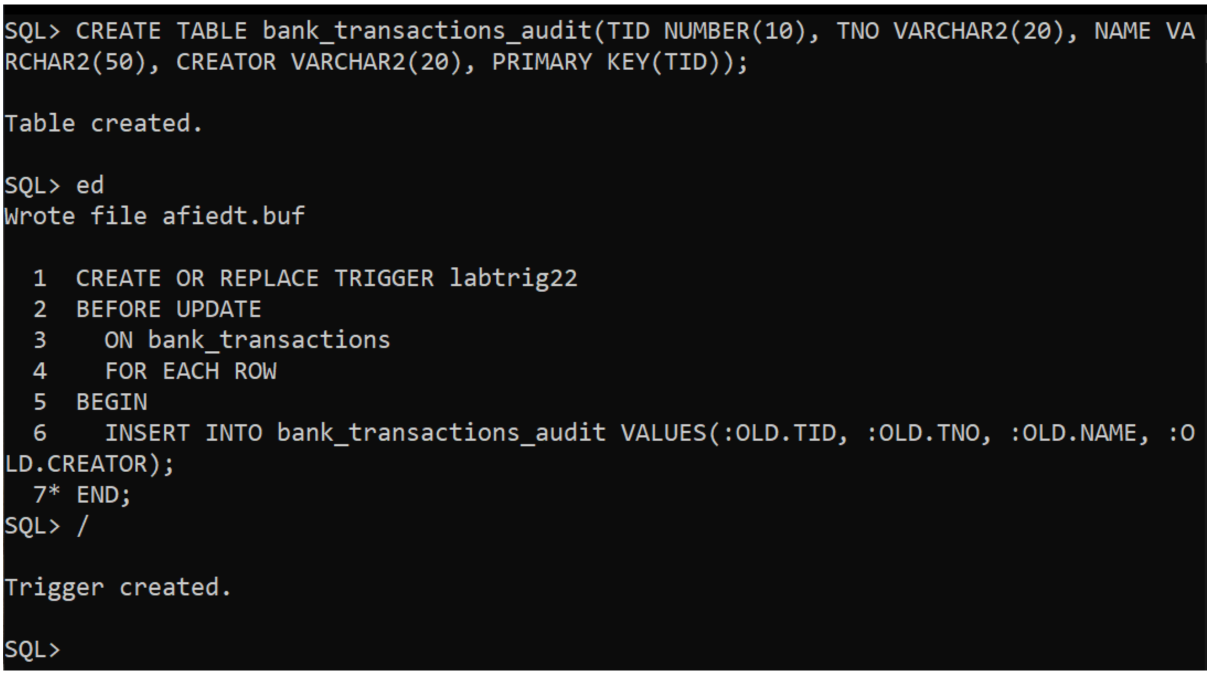


```
SQL> CREATE TABLE bank_transactions(TID NUMBER(10), TNO VARCHAR2(20), NAME VARCHAR2
(50), CREATOR VARCHAR2(20), PRIMARY KEY(TID));

Table created.

SQL> INSERT INTO bank_transactions VALUES('1','T1234', 'Peter', 'HR');

1 row created.

SQL> INSERT INTO bank_transactions VALUES('2','T9999', 'James', 'HR');

1 row created.

SQL>
```

Name : Siva Rama Krishna Nallapati
ID NO : 190031154

Q2. Create a table called ‘bank_transactions_audit’ with the same columns as ‘bank_transactions’. This will be the audit table. Then create a trigger ‘labtrig22’ which will work before updation in ‘bank_transactions’ table and create a copy of the record to be updated in the table ‘bank_transactions_audit’.

Sol)

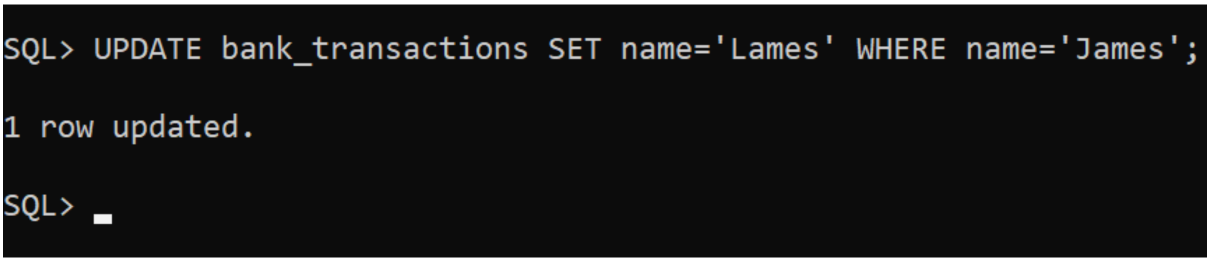


```
SQL> CREATE TABLE bank_transactions_audit(TID NUMBER(10), TNO VARCHAR2(20), NAME VA
RCHAR2(50), CREATOR VARCHAR2(20), PRIMARY KEY(TID));

Table created.

SQL> ed
Wrote file afiedt.buf

     CREATE OR REPLACE TRIGGER labtrig22
     BEFORE UPDATE
       ON bank_transactions
       FOR EACH ROW
     BEGIN
       INSERT INTO bank_transactions_audit VALUES(:OLD.TID, :OLD.TNO, :OLD.NAME, :O
LD.CREATOR);
     END;
SQL> /

Trigger created.

SQL>
```

Q3. Fire the trigger ‘labtrig22’ and test whether it works by updating the row where the name is ‘James’ and check the ‘bank_transactions_audit’ table and write the output. (That means you have to update the required row and display bank_transactions_audit to see if the updated row has been copied into the table successfully.

1. Update the required row.

UPDATE BANK_TRANSACTIONS SET NAME=’LAMES’ WHERE NAME='JAMES';

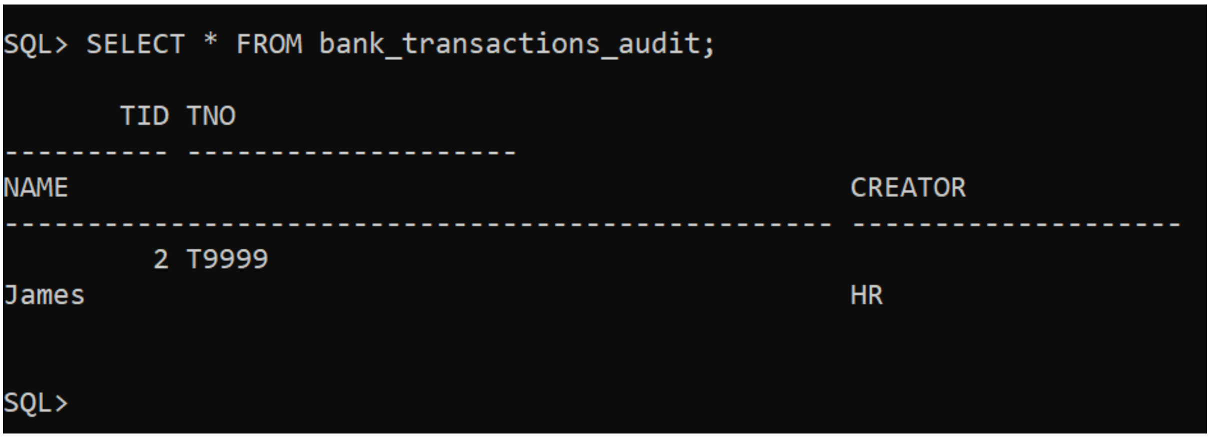


```
SQL> UPDATE bank_transactions SET name='Lames' WHERE name='James';

1 row updated.

SQL> _
```

Name : Siva Rama Krishna Nallapati
ID NO : 190031154

2. Display the ‘emp_backup_labtrig2’ table.

SELECT * FROM bank_transactions_audit;



```
SQL> SELECT * FROM bank_transactions_audit;

       TID TNO
---------- --------------------
NAME                                               CREATOR
-------------------------------------------------- --------------------
         2 T9999
James                                              HR


SQL>
```