--creating employee table--

CREATE TABLE employee (

emp\_id NUMERIC(10) NOT NULL,

emp\_name VARCHAR2(50) NOT NULL,

salary NUMBER(20),

dept\_name VARCHAR2(50),

CONSTRAINT emp\_pk PRIMARY KEY ( emp\_id )

);

--creating department table--

CREATE TABLE department (

dept\_id NUMERIC(10) NOT NULL,

dept\_name VARCHAR2(50) NOT NULL,

emp\_id NUMERIC(10) NOT NULL,

CONSTRAINT fk\_dept FOREIGN KEY ( emp\_id )

REFERENCES employee ( emp\_id )

);

--creating log data table--

CREATE TABLE log\_data

(

AUDIT\_ID NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,

TABLE\_NAME VARCHAR2(30),

TRANSACTION\_NAME VARCHAR2(10),

TRANSACTION\_DATE VARCHAR2(20),

EMP\_ID NUMBER(10),

EMP\_NAME VARCHAR2(20),

SALARY NUMBER(30),

DEPT\_NAME VARCHAR2(30),

DEPT\_ID NUMBER(20)

);

--creating procedure--

set serveroutput on;

CREATE OR REPLACE PROCEDURE emp\_dep\_disys(

emp\_id1 IN NUMBER,

emp\_name1 IN VARCHAR2,

salary1 IN NUMBER,

dept\_name1 IN VARCHAR2,

dept\_id1 IN NUMBER

) IS

e\_value NUMBER(20);

p\_value NUMBER(20);

ex\_invalid\_id EXCEPTION;

--CURSOR

CURSOR c1 IS

SELECT

emp\_id

FROM

employee

WHERE

emp\_id = emp\_id1;

BEGIN

OPEN c1;

FETCH c1 INTO e\_value;

IF e\_value <= 0 THEN

RAISE ex\_invalid\_id;

ELSE

IF ( c1%notfound ) THEN

INSERT INTO employee (

emp\_id,

emp\_name,

salary,

dept\_name

) VALUES (

emp\_id1,

emp\_name1,

salary1,

dept\_name1

);

INSERT INTO department (

dept\_id,

dept\_name,

emp\_id

) VALUES (

dept\_id1,

dept\_name1,

emp\_id1

);

COMMIT;

dbms\_output.put\_line('Value inserted successfully for: ' ||emp\_name1);

ELSE

--UPDATING the record

UPDATE employee

SET

emp\_name = emp\_name1,

salary = salary1,

dept\_name = dept\_name1

WHERE

emp\_id = e\_value;

UPDATE department

SET

dept\_id = dept\_id1,

dept\_name = dept\_name1

WHERE

emp\_id = e\_value;

COMMIT;

dbms\_output.put\_line('Value inserted successfully for: ' ||emp\_id1);

END IF ;

END IF;

CLOSE c1;

EXCEPTION

WHEN ex\_invalid\_id THEN

dbms\_output.put\_line('ID must be greater than zero!');

WHEN invalid\_number THEN

dbms\_output.put\_line('Datatype is invalid');

WHEN value\_error THEN

dbms\_output.put\_line('value is longer than the declared length ');

WHEN invalid\_cursor THEN

dbms\_output.put\_line('\*\*\*\*invalid cursor identified\*\*\*\*');

WHEN OTHERS THEN

raise\_application\_error(-20001,'error occured- '||SQLCODE||' -ERROR- '||sqlerrm);

END;

exec emp\_dep\_disys (1,'Pooja',500,'Dev',10);

exec emp\_dep\_disys (2,'Nairobi',2000,'IT',10);

exec emp\_dep\_disys (3,'Chloe',4000,'ITIS',10);

exec emp\_dep\_disys (1,'Pooja\_S',5000,'Dev',10);

CREATE OR REPLACE TRIGGER emp\_log AFTER

INSERT OR UPDATE OR DELETE ON employee

FOR EACH ROW

DECLARE

l\_transac VARCHAR2(10);

table\_name VARCHAR2(20);

BEGIN

CASE

WHEN inserting THEN

l\_transac := 'insert';

WHEN updating THEN

l\_transac := 'UPDATE';

WHEN deleting THEN

l\_transac := 'DELETE';

END CASE;

table\_name := 'employee';

INSERT INTO log\_data(

table\_name,

transaction\_name,

transaction\_date,

emp\_id,

emp\_name,

dept\_name

) VALUES (

table\_name,

l\_transac,

to\_char(sysdate, 'MM-DD-YYYY HH24:MI:SS'),

:new.emp\_id,

:new.emp\_name,

:new.dept\_name

);

END;

CREATE OR REPLACE TRIGGER dept\_log AFTER

INSERT OR UPDATE OR DELETE ON department

FOR EACH ROW

DECLARE

l\_value VARCHAR2(10);

table\_name VARCHAR2(20);

BEGIN

CASE

WHEN inserting THEN

l\_value := 'insert';

WHEN updating THEN

l\_value := 'UPDATE';

WHEN deleting THEN

l\_value := 'DELETE';

END CASE;

table\_name := 'department';

INSERT INTO log\_data (

table\_name,

transaction\_name,

transaction\_date,

emp\_id,

dept\_name,

dept\_id

) VALUES (

table\_name,

l\_value,

to\_char(sysdate, 'MM-DD-YYYY HH24:MI:SS'),

:new.emp\_id,

:new.dept\_name,

:new.dept\_id

);

END;