TRIGGERS:

A SET OF PL/SQL STATEMENTS STORED PERMANENTLY IN DATABASE AND "AUTOMATICALLY" ACTIVATED WHEN EVER AN EVENT RAISING STATEMENT (DML / DDL) IS PERFORMED.

THEY ARE USED TO IMPOSE USER DEFINED RESTRICTIONS(OR)BUSINESS RULES ON TABLE / SCHEMA.THEY ARE ALSO ACTIVATED WHEN TABLES ARE MANIPULATED BY OTHER USERS OR BY OTHER APPLICATION S/W TOOLS.THEY PROVIDE HIGH SECURITY ON TABLES.THEY ARE STORED IN "USER_TRIGGERS" SYSTEM TABLE.

1. DML TRIGGERS

2. DDL TRIGGER / DB TRIGGERS

PURPOSE OF TRIGGERS:

1. TO INVOKING USER DEFINED MESSAGE (OR) ALERTS AT EVENT RAISE.

2. TO CONTROL DML / DDL OPERATIONS.

3. TO IMPLEMENTING BUSINESS LOGICAL CONDITIONS.

4. TO VALIDATING DATA.

5. FOR AUDITING.

1. DML TRIGGERS:

- THESE TRIGGERS ARE EXECUTED BY SYSTEM AUTOMATICALLY WHEN USER PERFORM DML (INSERT / UPDATE / DELETE) OPERATIONS ON A SPECIFIC TABLE.

SYNTAX:

```
CREATE OR REPLACE TRIGGER <TRIGGER_NAME>
BEFORE/AFTER INSERT OR UPDATE OR DELETE
[ OF <COLUMNS>] ON <TABLE NAME>
[ FOR EACH ROW]
  WHEN <CONDITION> (TRUE -> EXECUTES THE TRIGGER,
                    FALSE - NOT EXECUTE)
  DECLARE
     <VARIABLE DECLARATION>;]
```

```
BEGIN
  <EXEC STATEMENTS>;
  [ EXCEPTION
      <EXEC STATEMENTS>;]
END;
```

TRIGGER EVENT:

- INDICATES WHEN TO ACTIVATE THE TRIGGER

BEFORE TRIGGER:

1. FIRST TRIGGER BODY IS EXECUTED
2. LATER DML COMMAND EXECUTED

AFTER TRIGGER:

1. FIRST DML COMMAND EXECUTED
2. LATER TRIGGER BODY EXECUTED

TRIGGER LEVELS:

- TRIGGER CAN CREATE AT TWO LEVELS.

A. ROW LEVEL TRIGGER:

- IN THIS LEVEL, TRIGGER BODY(LOGIC) IS EXECUTING FOR EACH ROW WISE FOR A DML OPERATION.

EX:

```
CREATE OR REPLACE TRIGGER TR1
AFTER UPDATE ON TEST
FOR EACH ROW
BEGIN
DBMS_OUTPUT.PUT_LINE('HELLO');
END;
/
```

TRIGGER CREATED.

TESTING:

SQL> UPDATE TEST SET SAL=10000 WHERE SAL=15000;

HELLO

HELLO

2 ROWS UPDATED.

B. STATEMENT TRIGGER:

- IN THIS LEVEL, TRIGGER BODY IS EXECUTING ONLY ONE TIME FOR A DML OPERATION.

EX:

```
CREATE OR REPLACE TRIGGER TR1
AFTER UPDATE ON TEST
BEGIN
DBMS_OUTPUT.PUT_LINE('HELLO');
END;
/
```

TRIGGER CREATED.

TESTING:

SQL> UPDATE TEST SET SAL=12000 WHERE SAL=10000;

HELLO

2 ROWS UPDATED.

BIND VARIABLES:

- BIND VARIABLES ARE JUST LIKE VARIABLES WHICH ARE USED TO STORE VALUES WHILE INSERTING, UPDATING, DELETING DATA FROM A TABLE.THESE ARE TWO TYPES,

1.: NEW:

- THIS BIND VARIABLE WILL STORE NEW VALUES WHEN WE INSERT.

SYNTAX:

: NEW. <COLUMN NAME>= <VALUE>;

EX:

: NEW.SAL = 15000;

2.: OLD:

- THIS BIND VARIABLE WILL STORE OLD VALUES WHEN WE DELETE.

SYNTAX:

:OLD. <COLUMN NAME>= <VALUE>;

EX:

: OLD.SAL = 12000;

NOTE: THESE BIND VARIABLES ARE USED IN "ROW LEVEL TRIGGERS ONLY".

1. TO INVOKING USER DEFINED MESSAGE / ALERT AT EVENT FIRE.

EX:

```
CREATE OR REPLACE TRIGGER TRINSERT
AFTER INSERT ON TEST
BEGIN
DBMS_OUTPUT.PUT_LINE ('SOME ONE INSERTED DATA INTO YOUR TABLE');
END;
/
```

TRIGGER CREATED.

TESTING:

SQL> INSERT INTO TEST VALUES (105,'SCOTT',36000);

SOME ONE INSERTED DATA INTO YOUR TABLE

1 ROW CREATED.

EX:

CREATE OR REPLACE TRIGGER TRUPDATE

AFTER UPDATE ON TEST

BEGIN

DBMS_OUTPUT.PUT_LINE ('SOME ONE UPDATED DATA INTO YOUR TABLE');

END;

/

TRIGGER CREATED.

TESTING:

UPDATE TEST SET SAL=22000 WHERE EID=1021;

SOME ONE UPDATING DATA IN YOUR TABLE

1 ROW UPDATED.

EX:

CREATE OR REPLACE TRIGGER TRDELETE

AFTER DELETE ON TEST

BEGIN

DBMS_OUTPUT.PUT_LINE ('SOME ONE DELETED DATA FROM YOUR TABLE');

END;

/

TRIGGER CREATED.

TESTING:

DELETE FROM TEST WHERE EDI=1022;

SOME ONE DELETING DATA FROM YOUR TABLE.

1 ROW DELETED.

EX:

```
CREATE OR REPLACE TRIGGER TRDML
AFTER INSERT OR UPDATE OR DELETE ON TEST
BEGIN
DBMS_OUTPUT.PUT_LINE ('SOME ONE PERFORMING DML OPERATIONS ON YOUR TABLE');
END;
/
```

TRIGGER CREATED.

2. TO CONTROL / RESTRICTED DML OPERATIONS ON A TABLE:

EX:

```
CREATE OR REPLACE TRIGGER TRIN
AFTER INSERT ON TEST
BEGIN
RAISE_APPLICATION_ERROR (-20487,'SOME ONE INSERTING DATA INTO YOUR TABLE');
END;
/
```

TESING:

```
INSERT INTO TEST VALUES (106,'MILLER',52000)
ERROR AT LINE 1:
ORA-20487: SOME ONE INSERTED DATA INTO UR TABLE
```

EX:

```
CREATE OR REPLACE TRIGGER TRUP
AFTER UPDATE ON TEST
BEGIN
RAISE_APPLICATION_ERROR (-20481,'SOME ONE UPDATING DATA IN YOUR TABLE');
END;
/
```

TESTING:

UPDATE TEST SET SAL=22000 WHERE EID=1021;

ERROR AT LINE 1:

ORA-20487: SOME ONE UPDATING DATA IN YOUR TABLE

EX:

CREATE OR REPLACE TRIGGER TRDEL

AFTER DELETE ON TEST

BEGIN

RAISE_APPLICATION_ERROR (-20481,'SOME ONE DELETING DATA FROM YOUR TABLE');

END;

/

TESTING:

DELETE FROM TEST WHERE EDI=1022;

ERROR AT LINE 1:

ORA-20487: SOME ONE DELETING DATA FROM YOUR TABLE.

EX:

CREATE OR REPLACE TRIGGER TRDEL

AFTER INSERT OR UPDATE OR DELETE ON TEST

BEGIN

RAISE_APPLICATION_ERROR (-20481,'SOME ONE PERFORMING DML OPERATIONS ON YOUR TABLE');

END;

/

TESTING:

INSERT INTO TEST VALUES (106,'MILLER',52000);

UPDATE TEST SET SAL=22000 WHERE EID=1021;

DELETE FROM TEST WHERE EDI=1022;

ERROR AT LINE 1:

ORA-20781: SOME ONE PERFORMING DML OPERATIONS ON YOUR TABLE.

3. TO IMPLEMENTING BUSINESS LOGICAL CONDITIONS:

EX:

CREATE A TRIGGER TO RESTRICTED DML OPERATIONS ON EVERY SATURDAY?

```
CREATE OR REPLACE TRIGGER TRDAY
AFTER INSERT OR UPDATE OR DELETE ON TEST
BEGIN
IF TO_CHAR(SYSDATE,'DY') = 'SAT' THEN
RAISE_APPLICATION_ERROR (-20456,'YOU CANNNOT PERFORM DML OPERATIONS ON EVERY SATURDAY');
END IF;
END;
/
```

EX:

CREATE A TRIGGER TO RESTRICTED DML OPERATIONS ON TEST TABLE IN BETWEEN 9AM TO 5PM?

```
CREATE OR REPLACE TRIGGER TRTIME
AFTER INSERT OR UPDATE OR DELETE ON TEST
BEGIN
IF TO_CHAR(SYSDATE,'HH24') BETWEEN 9 AND 16 THEN
RAISE_APPLICATION_ERROR (-20456,'YOU CANNNOT PERFORM DML OPERATIONS BETWEEN 9AM TO 5PM');
END IF;
END;
/
```

EX:

TRIGGER NAME: HOLI_TRIG

TABLE NAME: EMP

TRIGGER EVENT: BEFORE INSERT OR UPDATE OR DELETE

SOL:

```
CREATE OR REPLACE TRIGGER HOLI_TRIG
BEFORE INSERT OR UPDATE OR DELETE
ON EMP
DECLARE
 CNT NUMBER;
BEGIN
 IF TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 10 AND 16 THEN
    RAISE_APPLICATION_ERROR (-20001,'OFFTIMINGS, TRANS.
ARE NOT ALLOWED.');
 END IF;


 IF TO_CHAR(SYSDATE,'DY') IN ('SAT','SUN') THEN
    RAISE_APPLICATION_ERROR (-20002,'WEEKENDS, TRANS.
ARE NOT ALLOWED.');
 END IF;


 SELECT COUNT(HDATE) INTO CNT FROM HOLIDAY
        WHERE TO_CHAR(SYSDATE,'DD/MM/YY')
=TO_CHAR(HDATE,'DD/MM/YY');
 IF CNT>0 THEN
    RAISE_APPLICATION_ERROR (-20003,'TODAY PUBLIC
HOLIDAY, TRANS. ARE NOT ALLOWED.');
 END IF;
END;
```

4. TO VALIDATING DATA:

EX:

CREATE A TRIGGER TO VALIDATE INSERT OPERATION IF NEW SALARY IS LESS THAN 5000?

```
SQL> CREATE OR REPLACE TRIGGER TRSAL1
  BEFORE INSERT ON TEST
  FOR EACH ROW
  BEGIN
  IF: NEW.SAL < 5000 THEN
  RAISE_APPLICATION_ERROR (-20348,'NEW SAL SHOULD NOT BE LESS THAN TO 5000');
  END IF;
  END;
  /
```

TESTING:

INSERT INTO TEST VALUES (1021,'SMITH',4500); -----NOT ALLOWED

INSERT INTO TEST VALUES (1021,'SMITH',5500); -----ALLOWED

EX:

```
CREATE OR REPLACE TRIGGER DEPT_TRIG
BEFORE INSERT ON DEPT
FOR EACH ROW
BEGIN
 : NEW.DNAME: =UPPER (: NEW.DNAME);
 : NEW.LOC: =UPPER (: NEW.LOC);
END;
```

TESTING:

SQL> INSERT INTO DEPT VALUES (50,'ECONOMICS','HYD');

EX:

CREATE A TRIGGER TO VALIDATE UPDATE OPERATION ON TEST TABLE IF NEW SALARY IS LESS THAN TO OLD SALARY?

```
SQL> CREATE OR REPLACE TRIGGER TRSAL2
  BEFORE UPDATE ON TEST
  FOR EACH ROW
  BEGIN
  IF: NEW.SAL <: OLD.SAL THEN
  RAISE_APPLICATION_ERROR (-20748,'NEW SAL SHOULD NOT BE LESS THAN TO OLD SALARY');
  END IF;
  END;
  /
```

TESTING:

UPDATE TEST SET SAL=6000 WHERE SAL=8000; -----NOT ALLOWED

UPDATE TEST SET SAL=9000 WHERE SAL=8000; ------ ALLOWED

EX:

CREATE A TRIGGER TO VALIDATE DELETE OPERATION ON TEST TABLE IF WE TRY TO DELETE THE EMPLOYEE "SMITH" DETAILS?

```
SQL> CREATE OR REPLACE TRIGGER TRDATA
  BEFORE DELETE ON TEST
  FOR EACH ROW
  BEGIN
  IF: OLD.ENAME = 'SMITH' THEN
  RAISE_APPLICATION_ERROR (-20648,'WE CANNOT DELETE SMITH EMPLOYEE DETAILS');
  END IF;
  END;
  /
```

TESTING:

DELETE FROM TEST WHERE ENAME='SMITH’; -----NOT ALLOWED

DELETE FROM TEST WHERE ENAME='ALLEN’; ----ALLOWED

5. FOR AUDITING:

- WHEN WE MANIPULATE DATA IN A TABLE THOSE TRANSACTIONAL VALUES ARE STORED IN ANOTHER TABLE IS CALLED AS AUDITING TABLE.

EX:

SQL> CREATE TABLE EMP1(EID INT, ENAME VARCHAR2(10), SAL NUMBER (10));

TABLE CREATED.

SQL> CREATE TABLE AUDITEMP1(EID INT, AUDIT_INFOR VARCHAR2(100));

TABLE CREATED.

EX:

```
CREATE OR REPLACE TRIGGER TRAUDIT_INSERT
AFTER INSERT ON EMP1
FOR EACH ROW
BEGIN
INSERT INTO AUDITEMP1 VALUES (: NEW.EID,'SOME ONE INSERTED A NEW ROW INTO EMP1 TABLE ON'||' '||
TO_CHAR (SYSDATE,'DD-MON-YYYY HH:MI: SS AM'));
END;
 /
```

TESTING:

INSERT INTO EMP1 VALUES (1021,'SMITH',4500);

EX:

```
CREATE OR REPLACE TRIGGER TRAUDIT_UPDATE
AFTER UPDATE ON EMP1
FOR EACH ROW
BEGIN
INSERT INTO AUDITEMP1 VALUES (: OLD.EID,'SOME ONE
UPDATED DATA IN EMP1 TABLE ON'||' '||
TO_CHAR (SYSDATE,'DD-MON-YYYY HH:MI: SS AM'));
END;
 /
```

TESTING:

```
UPDATE EMP1 SET SAL=6000 WHERE EID=1021;
```

EX:

```
CREATE OR REPLACE TRIGGER TRAUDIT_DELETE
AFTER DELETE ON EMP1
FOR EACH ROW
BEGIN
INSERT INTO AUDITEMP1 VALUES (: OLD.EID,'SOME ONE
DELETED DATA FROM EMP1 TABLE ON'||' '||
TO_CHAR (SYSDATE,'DD-MON-YYYY HH:MI: SS AM'));
END;
 /
```

TESTING:

```
DELETE FROM EMP1 WHERE ENAME='SMITH';
```

EX:

```
CREATE OR REPLACE TRIGGER TRAUDIT_DML
AFTER INSERT OR UPDATE OR DELETE ON EMP1
FOR EACH ROW
BEGIN
INSERT INTO AUDITEMP1 VALUES (: OLD.EID,'SOME ONE
PERFORMING DML OPERATIONS ON EMP1 TABLE ON'||' '||
TO_CHAR (SYSDATE,'DD-MON-YYYY HH:MI: SS AM'));
END;
 /
```

TESTING:

```
INSERT INTO EMP1 VALUES (1021,'SMITH',4500);
UPDATE EMP1 SET SAL=6000 WHERE EID=1021;
DELETE FROM EMP1 WHERE ENAME='SMITH';
```

DDL TRIGGERS / DB TRIGGERS:

- THESE TRIGGERS ARE EXECUTED BY SYSTEM AUTOMATICALLY WHEN USER PERFORM DDL (CREATE / ALTER / DROP / RENAME) OPERATIONS ON A SPECIFIC SCHEMA / DATABASE.

- THESE TRIGGER ARE HANDLING BY DBA ONLY.

SYNTAX:

```
CREATE OR REPLACE TRIGGER <TRIGGER_NAME>
BEFORE/AFTER CREATE OR ALTER OR DROP OR RENAME
ON USERNAME.SCHEMA
[ FOR EACH ROW]
[ DECLARE
    <VARIABLE DECLARATION>;]
BEGIN
  <EXEC STATEMENTS>;
END;
```

EX:

```
SQL> CREATE OR REPLACE TRIGGER TRDDL
  AFTER CREATE ON MYDB9AM.SCHEMA
  BEGIN
  RAISE_APPLICATION_ERROR (-20456,'WE CANNOT CREATE A NEW TABLE IN MYDB9AM SCHEMA');
  END;
  /
TRIGGER CREATED.
SQL> CREATE TABLE T1(SNO INT);
ERROR AT LINE 1:
ORA-20456: WE CANNOT CREATE A NEW TABLE IN MYDB9AM SCHEMA
```

EX:

```
SQL> CREATE OR REPLACE TRIGGER TRDDL
  AFTER ALTER ON MYDB9AM.SCHEMA
  BEGIN
  RAISE_APPLICATION_ERROR (-20456,'WE CANNOT ALTER A TABLE IN MYDB9AM SCHEMA');
  END;
  /
SQL> ALTER TABLE EMP1 ADD EADD VARCHAR2(10);
ORA-20456: WE CANNOT ALTER TABLE IN MYDB9AM SCHEMA
SQL> CREATE OR REPLACE TRIGGER TRDDL
  AFTER DROP ON MYDB9AM.SCHEMA
  BEGIN
  RAISE_APPLICATION_ERROR (-20456,'WE CANNOT DROP A TABLE FROM MYDB9AM SCHEMA');
  END;
  /
TRIGGER CREATED.
```

SQL> DROP TABLE EMP1 PURGE;

ORA-20456: WE CANNOT DROP A TABLE FROM MYDB9AM SCHEMA

SQL> CREATE OR REPLACE TRIGGER TRDDL

AFTER RENAME ON MYDB9AM.SCHEMA

BEGIN

RAISE_APPLICATION_ERROR (-20456,'WE CANNOT RENAME A TABLE IN MYDB9AM SCHEMA');

END;

/

TRIGGER CREATED.

SQL> RENAME EMP1 TO EMPDETAILS;

ORA-20456: WE CANNOT RENAME A TABLE IN MYDB9AM SCHEMA

ORA-06512: AT LINE 2

EX:

SQL> CREATE OR REPLACE TRIGGER TRDDL

AFTER CREATE OR ALTER OR RENAME OR DROP ON MYDB9AM.SCHEMA

BEGIN

RAISE_APPLICATION_ERROR (-20456,'WE CANNOT PERFORM DDL OPERATIONS ON MYDB9AM SCHEMA');

END;

/

TRIGGER CREATED.

EX:

CREATE A TRIGGER TO RESTRICTED DDL OPERATIONS ON MYDB9AM SCHEMA IN BETWEEN 9AM TO 5PM?

```
CREATE OR REPLACE TRIGGER TRDDLTIME

AFTER CREATE OR ALTER OR RENAME OR DROP ON MYDB9AM.SCHEMA

BEGIN

IF TO_CHAR(SYSDATE,'HH24') BETWEEN 9 AND 16 THEN

RAISE_APPLICATION_ERROR (-20456,'YOU CANNNOT PERFORM DDL OPERATIONS ON MYDB9AM BETWEEN 9AM TO 5PM');

END IF;

END;

/
```

DROPPING TRIGGERS:

SQL> DROP TRIGGER <TRIGGERNAME>;

EX:

DROP TRIGGER TRDDLTIME;