PRACTICAL – 08

**NAME : Aryan Kashikar  
Roll n0: D2 32**

Practical 8: Triggers

AIM: To implement triggers in Oracle 11g

CREATE TABLE orders ( order\_id number(5), quantity number(4), cost\_per\_item number(6,2),

total\_cost number(8,2) );

CREATE TABLE order\_audit(username varchar2(20), auditdate date, order\_id number(5))

1. Create a trigger after delete on table order using condition total\_cost&gt;500. Trigger will be used

to take backup of orders deleted. ( Hint to take backup by creating backeup\_order table and

inserting data into that table using :old )

CREATE TABLE backup\_order AS

SELECT \* FROM orders WHERE total\_cost > 500;

CREATE OR REPLACE TRIGGER backup\_orders\_trigger

AFTER DELETE ON orders

FOR EACH ROW

WHEN (OLD.total\_cost > 500)

BEGIN

  INSERT INTO backup\_order(order\_id, quantity, cost\_per\_item, total\_cost)

  VALUES(:OLD.order\_id, :OLD.quantity, :OLD.cost\_per\_item, :OLD.total\_cost);

  COMMIT;

END;

2. create an AFTER INSERT trigger using the CREATE TRIGGER statement .

action performed by trigger will be insertion of :new values (i.e. username ,sysdate, orderid) in

table order\_audit.

(Hint: -- Finding username and date performing insert on the table :==

SELECT user ,sysdate INTO v\_username ,vdate FROM dual; )

CREATE OR REPLACE TRIGGER order\_insert\_audit\_trigger

AFTER INSERT ON orders

FOR EACH ROW

DECLARE

    v\_username VARCHAR2(20);

    v\_date DATE;

BEGIN

    SELECT USER, SYSDATE INTO v\_username, v\_date FROM dual;

    INSERT INTO order\_audit(username, auditdate, order\_id)

    VALUES(v\_username, v\_date, :NEW.order\_id);

    COMMIT;

END;

3. Create trigger before insert or update to check new value of total\_cots is less than 100. If it is

less than 100 then raise application error message “ not allowed less than 100”

CREATE OR REPLACE TRIGGER total\_cost\_check\_trigger

BEFORE INSERT OR UPDATE ON orders

FOR EACH ROW

BEGIN

    IF :NEW.total\_cost < 100 THEN

        RAISE\_APPLICATION\_ERROR(-20001, 'Not allowed: Total cost should be at least 100.');

    END IF;

END;

4. Create trigger after update that will display information of user, order\_id and date on which

order was updated after every update operation on table orders.

CREATE OR REPLACE TRIGGER order\_update\_info\_trigger

AFTER UPDATE ON orders

FOR EACH ROW

DECLARE

    v\_username VARCHAR2(20);

BEGIN

    SELECT USER INTO v\_username FROM dual;

    DBMS\_OUTPUT.PUT\_LINE('User: ' || v\_username);

    DBMS\_OUTPUT.PUT\_LINE('Order ID: ' || :NEW.order\_id);

    DBMS\_OUTPUT.PUT\_LINE('Date: ' || SYSDATE);

END;