Date _____ Day _____

"TRIGGERS"

→ A trigger is a user-defined SQL cmd that is invoked "automatically" in response to an event such as "insert, delete or update".

▷ SYNTAX :

```
Create Trigger
trigger_name trigger_time
trigger_event
On table_name For Each Row
Begin
.....
End;
```

trigger_time ⟶ before, after

trigger_event ⟶ insert, update, ddete

```
CREATE TRIGGER age_verify
BEFORE INSERT ON Customers
FOR EACH ROW
if new.age <0 then
   set new.age = 0;
endif;
```

Date _____ Day _____

1) DATA MANIPULATION LANGUAGE (DML) TRIGGERS :

→ Triggers that execute on before / After Insert, Update Delete.

```
CREATE TABLE Supesheroes (sh_name VARCHAR(15))
```

→ Before Insert Trigger

```
CREATE OR REPLACE TRIGGER bi_supesheroes
BEFORE INSERT ON Supesheroes
FOR EACH ROW
ENABLE
DECLARE
   v_user VARCHAR(15);
BEGIN
   SELECT user INTO v_user FROM dual;
   DBMS_OUTPUT.PUT_LINE ('You just inserted a row
                          Mr.' || v_user);
END; /
```

DUAL is a special one-row one-column table provided by oracle for dummy operations. It is often used when you need to select a value or perform a calculation without querying an actual table

user → built in oracle func that returns the current username

→ Before Update Trigger

```
CREATE OR REPLACE TRIGGER bu_supesheroes
BEFORE UPDATE ON Supesheroes
FOR EACH ROW
ENABLE
DECLARE
   v_user VARCHAR(5);
```

Date _____ Day _____

```
BEGIN
   SELECT user INTO v_user FROM dual;
   _____.
END; /
```

→ Insert, Update, Delete in a single trigger

```
CREATE OR REPLACE TRIGGER tr_superheroes
BEFORE INSERT OR DELETE OR UPDATE ON superh...
FOR EACH ROW
ENABLE
DECLARE
   v_user VARCHAR(15);
BEGIN
   SELECT user INTO v_user FROM Dual;

   IF INSERTING THEN
      DBMS_____('You are inserting');
ELSEIF DELETING THEN
      _____('You are deleting');
   IF UPDATING THEN
      _____('You are updating');
   END IF;

END; /
```

Page No. ☐

Date _____ Day _____

→ TABLE Auditing :

```
CREATE TABLE sh_audit (
    New_name ___
    Old_name ___
    User_name ___
    Entry_date ___
    Operation ___
);

CREATE OR REPLACE TRIGGER Super_audit
BEFORE INSERT OR DELETE OR UPDATE ON Superheroes
FOR EACH ROW ENABLE
DECLARE
    v_user varchar(15);
    v_date varchar(30);
BEGIN
SELECT user, TO_CHAR(sysdate, 'DD/MM/YYYY HH24:MI:SS')
INTO v_user, v_date FROM dual;

IF INSERTING THEN
  INSERT INTO sh_audit(new_name, old_name,
                       username, entrydate, oper)
VALUES (:NEW.sh_name, NULL, v_user, v_date, 'insert');
```

Date _____

```
ELSEIF DELETING THEN
    INSERT INTO sh_audit ( ____________ )
    VALUES (NULL; :OLD.sh_name, v_user, v_da )

ELSEIF UPDATING THEN
    INSERT INTO sh_audit( ____________ )
    VALUES (:NEW.sh_name, :OLD.sh_name, ____ );

ENDIF;

END; /
```

NOTE:

- For an INSERT TRIGGER
  - ↳ OLD Contains no values
  - ↳ NEW Contains the new values
- For an UPDATE TRIGGER
  - ↳ OLD contains old val
  - ↳ NEW contain new val
- For a DELETE TRIGGER
  - ↳ OLD contains old val
  - ↳ NEW contains no values.

Day

DDL TRIGGERS :

→ Schema Auditing

```
CREATE TABLE schema_audit (
    Ddl_date  —
    Ddl_user  —
    Object_created  —
    Object_name  —
    Ddl_operation  —
);
```

→ This table should be made in same ~~table~~ schema that needs to be audited.

```
CREATE OR REPLACE TRIGGER hr_audit_tr
AFTER DDL ON SCHEMA
BEGIN
  INSERT INTO SCHEMA_Audit
  VALUES (sysdate, sys_context('USERENV', 'CURRENT_USER'),
          ora_dict_obj_type, ora_dict_obj_name,
          ora_sysevent);
END; /
```

→ Database Auditing

For database auditing it is necessary to login to the DB using either SYS user or SYSTEM user.

The code for trigger is same the only difference is instead of SCHEMA write DATABASE

Page No.

event type → ora. sys even

Date Day

database events can be → LOGON, LOGOFF, DDL
STARTUP, SHUTDOWN

"VIEWS"

→ It is a virtual table
→ View is the result of stored query.
→ The view is not stored in physical space
→ The table from which the view is taken is called base table the record of the
→ Any change done in base table also reflected on views.
→ Two types of views.
  ↳ Read Only → You cannot update, insert, delete
  ↳ Updatable views → Any updation, deletion, insertion done on view also reflected on base table.
→ DDL cmds cannot be applied to a view
→ A single view can contain data of more than one table.

```
OR REPLACE
CREATE VIEW customer_names AS
SELECT id, names
FROM customers
```

Table name → ORA_DICT_OBJ_Name
TYPE → ORA_DICT_OBJ_Type
RAISE_APPLICATION_ERROR(-2001, -20Day

Date

→ CREATE OR REPLACE → if there is not a view it will create it or else it will replace the existing one.

Rules:

↳ Cannot change col name (means ags je phele view banya tha usk name cust_id tha to new view ka customer_id nhi rakh sakte)
↳ Cannot change col data type
↳ Cannot change order of columns
↳ Can add a new col in the end.
↳ The conditions can be change like WHERE, JOIN etc.
↳ But the structure cannot be changed.

To change the column name

```
ALTER VIEW customer_names
RENAME column id to cust_id;
```

Droping the view

```
DROP view customer_names;
```

If the structure of the base table is changed the change is not reflected on the view.
Because the structure of the view had already been saved when the view was created.

Date _____ Day _____

→ Views that are only created from a single table are updatable.
↳ It cannot have distinct clause.
↳ It cannot have GROUP BY clause.
↳ " " " WITH clause
↳ " " " WINDOW Function

} if a view is using any of these it cannot be updated.

→ CREATE OR REPLACE VIEW apple_products
AS
SELECT * FROM products where brand = 'Apple'
with check option ;
↳ this ensure that only apple products are added.

Insert into apple_products
values ('P22', 'Note 22', 'Samsung', 2500, NULL);
↳ this data cannot be inserted because brand name is samsung. Only data having brand 'Apple' can be inserted.