



---

# COMPTE RENDU TP6

---



**BEN OSMANE HICHAM**

ENSAH 2020-2021

## EXERCICE 1:

1. Creation de table ARTICLES:

**CREATE TABLE ARTICLES (nom\_Prod VARCHAR2(255), Quantite NUMBER, Prix\_u number, cout number);**

2. Insérer la ligne suivante nom\_prod= LapTopHP , Quantite=5, prix=3600,00

3.

The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the editor is as follows:

```
CREATE TABLE ARTICLES (nom_Prod VARCHAR2(255), Quantite NUMBER, Prix_u number, cout number);

INSERT INTO ARTICLES(nom_Prod,Quantite,Prix_u) VALUES('LapTopHP',5,3600.00);
commit;

select * from ARTICLES;
```

Below the script, the 'Query Result' tab shows the execution results. The status bar indicates 'All Rows Fetched: 2 in 0.069 seconds'. The results are displayed in a table with the following columns: NOM\_PROD, QUANTITE, PRIX\_U, and COUT.

	NOM_PROD	QUANTITE	PRIX_U	COUT
1	LapTopHP	5	3600	(null)

4. Creation de trigger:

The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the editor is as follows:

```
CREATE OR REPLACE TRIGGER ARTICLE_TRIG1
BEFORE
INSERT OR UPDATE ON ARTICLES
FOR EACH ROW BEGIN
:NEW.COUT := :NEW.QUANTITE* :NEW.Prix_u;
END;
```

Below the script, the 'Query Result' tab shows the execution results. The status bar indicates 'Task completed in 0.029 seconds'. The results are displayed in a table with the following columns: NOM\_PROD, QUANTITE, PRIX\_U, and COUT.

	NOM_PROD	QUANTITE	PRIX_U	COUT
1	LapTopHP	5	3600	(null)

## 5. Tester le trigger

### 1) Insertion de nouveau lines

```
INSERT INTO ARTICLES (nom_Pod,Quantite,Prix_u) VALUES('Lap_Top_COM',10,2000.5);
INSERT INTO ARTICLES (nom_prod,quantite,prix_u) VALUES('Mouse_King',40,20.00);
INSERT INTO ARTICLES (nom_prod,quantite,prix_u) VALUES('USB_Key_Row',60,23.00);
INSERT INTO ARTICLES (nom_Prod,quantite,prix_u) VALUES('Lap_Top_Del',2,2300.00);
commit;
select * from ARTICLES;
```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.002 seconds

	NOM_PROD	QUANTITE	PRIX_U	COUT
1	LapTopHP	5	3600	(null)
2	LapTopHP	5	3600	(null)
3	Mouse_King	40	20	800
4	USB_Key_Row	60	23	1380
5	Lap_Top_Del	2	2300	4600

2) Après l'insertion des données le trigger ARTCILE\_TRG1 déclenche automatiquement pour calculer le cout de chaque produit ajouter.

3) en une seule instruction, Procéder à une mise à jour du prix de tous les articles Lap\_Top à 2500, Que est ce que vous constater pour le champ cout

```
UPDATE ARTICLES SET Prix_u=2500 WHERE nom_Prod LIKE '%Lap%Top%';
commit;
```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.002 seconds

	NOM_PROD	QUANTITE	PRIX_U	COUT
1	LapTopHP	5	2500	12500
2	LapTopHP	5	2500	12500
3	Mouse_King	40	20	800
4	USB_Key_Row	60	23	1380
5	Lap_Top_Del	2	2500	5000

6. Vérifier que le trigger ARTCILE\_TRG1 est actif.

**alter trigger ARTCILE\_TRG1 enable;**

The screenshot shows a SQL Developer window with a query editor at the top containing the command: `select status from all_triggers where trigger_name like 'ARTICLE_TRG1'`. Below the editor, the 'Script Output' pane displays the following messages: '3 rows updated.', 'Commit complete.', and a table showing the status of the trigger.

STATUS
ENABLED

a-Désactiver le trigger et essayer d’insérer un nouveau enregistrement ou de MAJ la table ARTCILES, Que est ce que vous constatez pour le champ cout?

**alter trigger ARTCILE\_TRG1 disable;**

The screenshot shows a SQL Developer window with a query editor at the top containing the command: `ALTER TRIGGER ARTICLE_TRG1 DISABLE;`. Below the editor, the 'Script Output' pane displays the following messages: 'Commit complete.', a table showing the status of the trigger, and 'Trigger ARTICLE\_TRG1 altered.'.

STATUS
ENABLED

```
INSERT INTO ARTICLES(nom_Prod, quantite, prix_u) VALUES('USB_COM',50,60);
commit;
select * from ARTICLES;
```

Script Output x

Query Result x

SQL

All Rows Fetched: 6 in 0.002 seconds

	NOM_PROD	QUANTITE	PRIX_U	COUT
1	LapTopHP	5	2500	12500
2	LapTopHP	5	2500	12500
3	Mouse_King	40	20	800
4	USB_Key_Row	60	23	1380
5	Lap_Top_Del	2	2500	5000
6	USB_COM	50	60	(null)

Le champ cout est null par ce qu'on a désactiver le trigger qui permet de calculer le cout d'un produit automatiquement

b-Activer le de nouveau et essayer d'insérer un nouveau enregistrement ou de MAJ la table ARTICLES, Que est ce que vous constater pour le champ cout ?

```
ALTER TRIGGER ARTICLE_TRIG1 ENABLE;
UPDATE ARTICLES SET Prix_u=Prix_u;
commit;
select * from ARTICLES;
```

Script Output x

Query Result x

NOM_PROD	QUANTITE	PRIX_U	COUT
1 LapTopHP	5	2500	12500
2 LapTopHP	5	2500	12500
3 Mouse_King	40	20	800
4 USB_Key_Row	60	23	1380
5 Lap_Top_Del	2	2500	5000
6 USB_COM	50	60	3000

## EXERCICE 2:

```

CREATE OR REPLACE TRIGGER EMP_TRG1
BEFORE UPDATE OF SALARY ON EMPLOYEES
FOR EACH ROW
DECLARE User_non_autorise EXCEPTION;
BEGIN
IF (USER<>'HR') THEN
RAISE User_non_autorise;
END IF
EXCEPTION WHEN User_non_autorise THEN
RAISE_APPLICATION_ERROR(-20001, 'SEUL le MANAGER peut CHANGER le Salaire');
END;

```

1. Tester le trigger
1. faire un update de salaire d'un employé de votre choix. update employeesset salary = 100 where employee\_id = 104;

1 ligne mis à jour.

1.b) créer un autre utilisateur TEST\_ENSAH

```
SQL> CREATE USER TEST_ENSAH IDENTIFIED BY 12345 ACCOUNT UNLOCK;

User created.

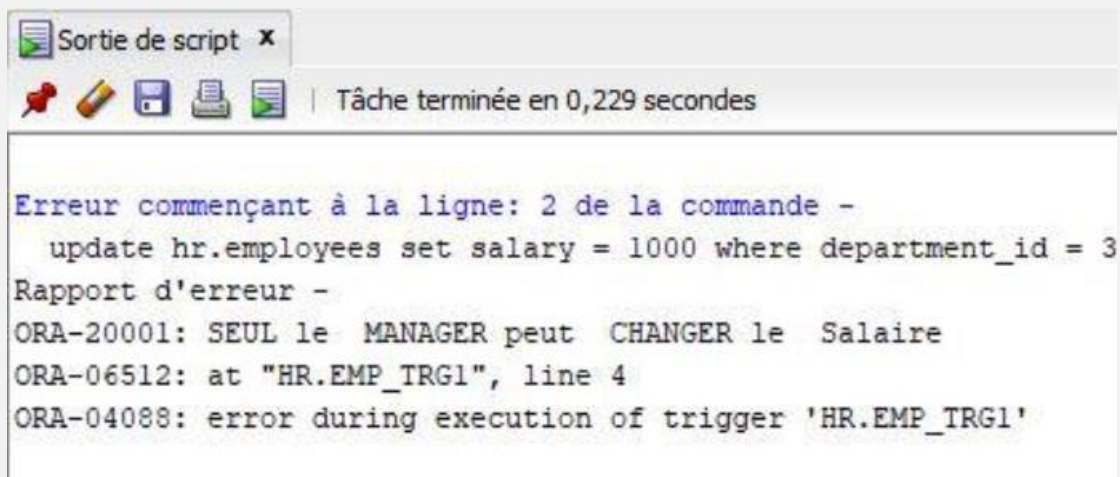
SQL> Grant connect,resource to TEST_ENSAH;

Grant succeeded.

SQL> GRANT select, insert, update on EMPLOYEES to TEST_ENSAH;
GRANT select, insert, update on EMPLOYEES to TEST_ENSAH
```

1.c) se connecter en tant qu'un autre utilisateur et faire la MAJ de salaire des employés de département 30. Qu'est ce que vous constatez ?

Update hr.employees set salary = 1000 where department\_id = 30;



On remarque que même si on a donné les droits de modification sur la table employees à un autre utilisateur le salaire ne peut être modifié. Le trigger EMP\_TRG1 a interdit de faire une modification avec un autre utilisateur.

2-Désactiver le trigger et faire le même 'UPDATE. Qu'est-ce que vous constatez ?

alter trigger EMP\_TRG1 disable;

update hr.employees set salary = 1000 where department\_id = 30;



La modification a bien été effectuée car on a désactivé le trigger EMP\_TRG1.

### EXERCICE 3:

1. Créer la table EVALUATION(id, note\_DS1, note\_DS2, note\_DSTP, Moyenne) tous ces champs sont de type number, la colonne moyenne est la moyenne des notes des 3 DS.

```
CREATE TABLE EVALUATION
(
  IDNUMBER,
  NOTE_DS1NUMBER(4,2),
  NOTE_DS2NUMBER(4,2),
  NOTE_DSTPNUMBER(4,2),
  MOYENNENUMBER(4,2),
  PRIMARY KEY (ID)
);
```

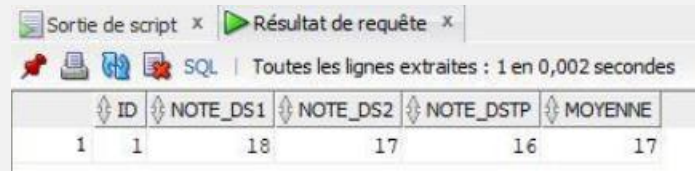
1-On veut créer un trigger Evalu\_TRG pour calculer la moyenne quelles sont les order MLD à prévoir pour ce trigger.

2-Créer le trigger Evalu\_TRG

```
CREATE OR REPLACE TRIGGER EVALU_TRI
before insert or update or delete onevaluation
for each row
begin
if INSERTING or UPDATING then
:NEW.Moyenne := (:new.note_ds1 + :new.note_ds2 + :new.note_dstp) / 3;
ELSIF DELETING then
dbms_output.put_line('Etudiant supprimé avec succès');
end if;
end;
```

3-Tester ce trigger avec les order MLD concernées.

**insert into evaluation(ID,NOTE\_DS1,NOTE\_DS2,NOTE\_DSTP) values(1,18,17,16);**

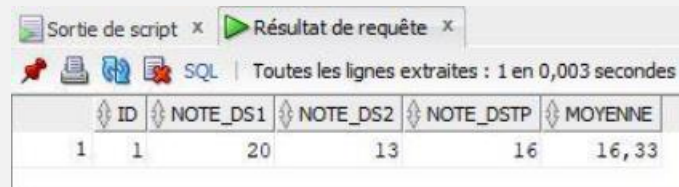


Sortie de script x Résultat de requête x

Toutes les lignes extraites : 1 en 0,002 secondes

ID	NOTE_DS1	NOTE_DS2	NOTE_DSTP	MOYENNE
1	18	17	16	17

Update evaluation set note\_ds1 = 20,note\_ds2 = 13 where id = 1;



Sortie de script x Résultat de requête x

Toutes les lignes extraites : 1 en 0,003 secondes

ID	NOTE_DS1	NOTE_DS2	NOTE_DSTP	MOYENNE
1	20	13	16	16,33