

# TP1 : PL/SQL (séance sécurité des schémas)

## 1. Schéma de base de données

---

Nous allons travailler sur le schéma relationnel Employé (schéma de test classique pour Oracle). Le script de création des tables vous est donné (ScriptCreation.sql).

- Dept(**n\_dept**, nom, lieu)
- Emp(nom, **num**, fonction, n\_sup, embauche, salaire, com, n\_dept)  
avec  $\text{Emp}(n\_dept) \subseteq \text{Dept}(n\_dept)$   
et  $\text{Emp}(n\_sup) \subseteq \text{Emp}(num)$

## 2. Contraintes clés primaires et étrangères

---

Vous exécuterez le script de création une fois connecté à Oracle (BD master).

```
Exemple pour le compte exxxxxx et le mdp yyyy :  
rlwrap sqlplus exxxxxx/yyyy@oracle.etu.umontpellier.fr:1523/pmaster  
  
Si pas de compte ou pb :  
aller sur  
https://sapiens.umontpellier.fr/
```

Listing 1 – connexion

Vous définirez les contraintes de clés comme indiquées dans le schéma relationnel, au moyen d'un ordre ALTER TABLE. Pensez à exploiter le "on delete cascade" pour les contraintes de clé étrangère, qui permettra par exemple de supprimer les tuples des employés qui travaillent dans un département qui n'existe plus.

```
alter table emp add constraint emp_pk primary key (num);  
alter table dept add constraint dept_pk primary key (n_dept);  
alter table emp add constraint emp_fk1 foreign key (n_sup) references emp(num) on delete  
cascade;  
alter table emp add constraint emp_fk2 foreign key (n_dept) references dept(n_dept) on  
delete cascade;
```

Listing 2 – ajout de contraintes

## 3. Triggers LMD

---

Une partie des questions est à traiter en non présentiel.

1. Construisez un trigger nommé *rennesMille* qui vérifie que le salaire des employés est toujours supérieur à 1000 euros, à l'insertion comme à la mise à jour, mais uniquement pour des employés qui travaillent dans le département localisé à Rennes.
2. Les triggers ont souvent une écriture très simple (règle ECA) et s'appuient sur des procédures qui en masquent la complexité. Vous reprendrez l'écriture du trigger ouvrable (donné dans le cours). Vous définirez une procédure *JoursEtHeuresOuvrables* sans argument qui vérifie que la date du jour n'est pas un samedi ni un dimanche et qui renvoie un message d'erreur autrement. Vous redéfinirez le trigger *ouvrable* qui fera appel à cette procédure, dans le contexte de la table *Emp*. Vous en testerez les effets.
3. Les triggers ont souvent un rôle de monitoring auprès des administrateurs de bases de données (indicateurs de la bonne santé des bases de données dont ils ont la charge). Vous créerez une table historique (*dateOperation*, *nomUsager*, *typeOperation*) qui va permettre de conserver la trace de toutes les opérations réalisées sur la table *Dept*. Vous créerez le trigger nommé *monitor* qui va permettre d'alimenter cette table.  
Le nom de l'utilisateur et la date système sont connus au travers des descripteurs *USER* et *SYS-DATE* (cf. `select user, sysdate from dual;`)
4. Lors de la définition de contraintes de clés étrangères, Oracle autorise certaines fonctionnalités telle que la suppression en cascade des tuples dépendants au travers de la syntaxe réservée *on delete cascade*. A la différence de PostgreSQL, Oracle ne permet pas l'exploitation de la syntaxe *on update cascade* qui permettrait de modifier les tuples dépendants en conséquence.  
Vous construirez un trigger nommé *cascade* qui porte sur la table *Dept* et qui se charge à chaque événement de suppression ou de modification d'un département (*n\_dept*) dans *Dept* de supprimer ou de modifier dans la table *Emp*, les tuples d'employés dépendants de ce département.  
**Pensez à supprimer au préalable la contrainte de clé étrangère sur *n\_dept* de la table (pose des problèmes de conflit sinon). Pensez ensuite à annuler les effets des suppressions ou modifications par rollback sur la transaction.**

## 4. Triggers LDD

---

Vous construirez un trigger qui se déclenche au niveau de votre schéma utilisateur et qui affiche un message indiquant un changement du modèle à chaque ordre de création d'objet (table ou autre objet du schéma).

## 5. Triggers d'instance

---

Vous construirez un trigger qui se déclenche au niveau de chaque connexion utilisateur (événement qui se produit au niveau de la base de données). L'action découlant de cet événement devra être une insertion dans une table nommée *QuiSeConnecte* que vous aurez préalablement définie. Prenez garde de définir des tailles de chaîne variable suffisamment élevées pour la variable *OS\_USER* (au moins 60 caractères). La définition de cette catégorie de trigger nécessite le privilège : "administrer database trigger", et donc l'octroi de ce privilège par le DBA, de la manière suivante :

```
grant administrer database trigger to public;
```

Vous utiliserez la fonction *sys\_context* qui donne des informations sur l'environnement de l'utilisateur et qui peut être exploitée à partir de la table à tout faire *dual*.

```
select sys_context('USERENV','IP_ADDRESS') from dual ;
select sys_context('USERENV','SESSION_USER') from dual ;
```

## 6. Correction

---

Il s'agit d'exemples de correction. Vous pouvez choisir de traiter les questions différemment.

### 6.1 LMD

#### 6.1.1 Question 1

À l'aide d'un curseur implicite pour le numéro de département de Rennes.

```
create or replace trigger rennesMille
after insert or update of salaire on emp
for each row
declare
n_d dept.n_dept%type;
begin
select n_dept into n_d from dept where lieu = 'Rennes';
if :new.n_dept=n_d and :new.salaire <1000
then raise_application_error(-20100,'forbidden');
end if;
exception
when no_data_found then raise_application_error(-20001,'aucun departement a Rennes
'||SQLCODE);
when too_many_rows then raise_application_error(-20002,'plusieurs departement a Rennes
'||SQLCODE);
when others then raise_application_error(-20003,'survenue du probleme avec le code et le
message : '||SQLCODE||' '||SQLERRM);
end;
/

--exemple evenement pour test du declenchement
insert into emp (nom,num,salaire,n_dept) values ('bob',3345,500,10);
```

Listing 3 – correction Q1

#### 6.1.2 Question 2

Mise à contribution de la fonction to\_char avec un filtre exemple (DY : 3 premières lettres du jour en majuscules)

```
--retourner date jour
select to_char(sysdate,'DY') from dual;

create or replace procedure ouvrable is
begin
if trim(to_char(sysdate,'DY'))='JEU.'
then raise_application_error(-20100,'forbidden');
end if;
end;
```

```
/
create or replace trigger t_ouvrable
after insert or delete or update on emp
begin
ouvrable;
end ;
/

insert into emp (nom,num,salaire,n_dept) values ('bob',3375,1500,10);
alter trigger t_ouvrable disable;
```

Listing 4 – correction Q2

### 6.1.3 Question 3

```
create table historique (dateOperation date, usager varchar(20), typeOperation
    varchar(20));

create or replace trigger monitor
before insert or update or delete on dept
for each row
begin
if inserting then insert into historique values (sysdate, user,'insertion');
elsif deleting then insert into historique values (sysdate,user,'suppression');
else insert into historique values (sysdate, user, 'modification');
end if;
end;
/

insert into dept values (200,'certification', 'Montpellier');
update dept set lieu = 'Lunel' where n_dept = 200;
delete from dept where n_dept = 200;

select * from historique;
```

Listing 5 – correction Q3

### 6.1.4 Question 4

```
create or replace trigger t_updateCascade
after delete or update on dept
for each row
begin
if deleting
then delete from emp where n_dept = :old.n_dept;
elsif updating then update emp set n_dept = :new.n_dept where n_dept = :old.n_dept;
end if;
end ;
/

insert into emp (nom,num,salaire,n_dept) values ('zoe',4375,1500,20);

update dept set n_dept=100 where n_dept = 20;
-- voir les effets puis annuler la transaction par rollback ;
```

```
-- veuillez a supprimer la contrainte de cle etrangere au prealable sous peine de conflit
```

Listing 6 – correction Q4

## 6.2 LDD

### 6.2.1 Question 1

```
set serveroutput on
create or replace trigger creation before create on Exxx.schema
begin
dbms_output.put_line('creation aboutie le '||to_char(sysdate,'DAY'));
end ;
/

alter trigger creation disable;
```

Listing 7 – correction Q1

## 6.3 Instance

### 6.3.1 Question 1

```
DROP TABLE QuiSeConnecte;
CREATE TABLE QuiSeConnecte
(
c_user varchar2(60),
os_user varchar2(60),
ip_address varchar2(20),
c_date date
);

CREATE OR REPLACE TRIGGER logon_db
after logon ON DATABASE
declare
o_user QuiSeConnecte.os_user%type ;
i_address QuiSeConnecte.ip_address%type ;
session_user QuiSeConnecte.c_user%type ;

begin
select sys_context('USERENV','OS_USER') into o_user from dual ;
select sys_context('USERENV','SESSION_USER') into session_user from dual ;
select sys_context('USERENV','IP_ADDRESS') into i_address from dual ;
INSERT INTO QuiSeConnecte VALUES (session_user, o_user,i_address,sysdate);
commit;
end;
/

select c_user, os_user, ip_address, TO_CHAR(c_date,'YYYY-MM-DD HH24:MI:SS')
from quiseconnecte;
```

Listing 8 – correction Q1