ASGBD

Rapport du TP N°5: « Triggers »

CHIKH Khadidja

Master 1 SII
Groupe 4

02/11/2018

1) Créer un trigger qui affiche « un nouveau employé de type infirmier est ajouté» après chaque insertion d'un infirmier. Répéter la même chose pour la modification ou la suppression.

Nous utilisons la syntaxe donnée. D'abord le trigger qui informe de l'*insertion* :
Nous utilisons le mot «after» pour dire que ça doit se passer après l'opération (de l'événement déclencheur), et «insert» pour dire que l'évènement déclencheur est l'insertion, « on » sur la table « Infirmier » et « for each row » à chaque fois qu'il y est insertion.
(Nous appliquons le même principe dans les questions qui suivent)

```
SQL> create or replace trigger ins_inf after insert on infirmier for each row
  2  begin
  3  dbms_output.put_line('Un nouveau employe de type infirmier est ajoute!');
  4  end;
  5  /

Trigger created.
```

En executant une insertion, nous obtenons les résultats suivants :

```
SQL> insert into employe(num_emp)values(255);

1 row created.

SQL> insert into infirmier values(255,'CHG','jour',40000);
Un nouveau employe de type infirmier est ajoute!

1 row created.
```

Le message est affiché après l'insertion d'un nouveau infirmier.

Cas de *modification :*

```
SQL> create or replace trigger mdf_inf after update on infirmier for each row
  2  begin
  3  dbms_output.put_line('Les informations d un infirmier ont été mises à jour');
  4  end;
  5  /

Trigger created.
```

Exemple d'execution :

```
SQL> update infirmier set code_service='CAR' where num_inf=255;
Les informations d un infirmier ont été mises à jour

1 row updated.
```

Cas de *suppression:*

```
SQL> create or replace trigger sup_inf after delete on infirmier for each row
  2  begin
  3  dbms_output.put_line('Un infirmié supprimé!');
  4  end;
  5  /

Trigger created.
```

Exemple d'execution :

```
SQL> delete from infirmier where num_inf=255;
Un infirmié supprimé!

1 row deleted.
```

2) Créer un trigger qui affiche « un nouveau infirmier est affecté à un [Nom de service] » après chaque insertion d'un infirmier.

Code du trigger :

```
SQL> create or replace trigger ins_inf after insert on infirmier for each row
  2  begin
  3  dbms_output.put_line('Un nouveau infirmié est affecté au ' ||:new.code_service);
  4  end;
  5  /

Trigger created.
```

Exemple d'execution :

```
SQL> insert into employe(num_emp)values(256);

1 row created.

SQL> insert into infirmier values(256,'CHG','jour',40000);
Un nouveau infirmié est affecté au CHG

1 row created.
```

Nous voyons que le trigger affiche le message voulu après l'insertion d'un nouveau infirmier dans le service CHG.

3) Créer un trigger qui vérifie avant modification du code_service dans la table infirmier que la nouvelle valeur existe réellement, sinon, il refuse l'opération.

Nous ajoutons le «of code_service» pour spécifier que le déclenchement ne se passe que quand il s'agit de la modification de cette colonne. Nous utilisons la procédure *raise_application_error* dans le bloc de l'exception, pour empêcher la modification en cas d'erreur:

```
SQL> create or replace trigger mdf_inf_srv before update of code_service on infirmier f
or each row
  2      declare
  3      n number(2);
  4      cs varchar2(3);
  5      e exception;
  6      begin
  7      cs:=:new.code_service;
  8      select count(*) into n from service where code_service=cs;
  9      if(n=0) then raise e;
 10      else dbms_output.put_line('L infirmier n° '||:new.num_inf||',affecté au '||:ol
d.code_service||' est reaffecté au '||cs||'!');
 11     end if;
 12     EXCEPTION
 13     when e then raise_application_error(-20006,'Opération refusée: code de service
entré invalide!');
 14     end;
 15     /

Trigger created.
```

Exemple d'execution :

```
SQL> update infirmier set code_service='CAR' where num_inf=256;
L infirmier nº 256,affecté au CHG est reaffecté au CAR!

1 row updated.

SQL> update infirmier set code_service='CxR' where num_inf=256;
update infirmier set code_service='CxR' where num_inf=256
       *
ERROR at line 1:
ORA-20006: Opération refusée: code de service entré invalide!
ORA-06512: at "DBAHOPITAL.MDF_INF", line 12
ORA-04088: error during execution of trigger 'DBAHOPITAL.MDF_INF'
```

Le trigger arrête l'opération et affiche effectivement le message personnalisé auparavant quand nous essayons d'insérer un infirmier avec une valeur de code_service qui n'existe pas.

4) Créer un trigger qui vérifie que lors de la modification du salaire d'un infirmier, la nouvelle valeur ne peut jamais être inférieure à la précédente.

Code du trigger:

```
SQL> create or replace trigger mdf_inf before update of salaire on infirmier for each r
ow
  2  declare
  3  e exception;
  4  begin
  5  if(:new.salaire<:old.salaire) then raise e;
  6  else dbms_output.put_line('L infirmier nº '||:new.num_inf||',son ancien salaire es
t:'||:old.salaire||'DA et son nouveau salaire est: '||:new.salaire||'DA.');
  7  end if;
  8  EXCEPTION
  9  when e then raise_application_error(-20006,'Opération refusée: le nouveau salaire
doit être supérieur à celui d avant!');
 10  end;
 11  /

Trigger created.
```

Exemple d'execution :

```
SQL> update infirmier set salaire=50000 where num_inf=256;
L infirmier nº 256,son ancien salaire est:40000DA et son nouveau salaire est:
50000DA.

1 row updated.

SQL> update infirmier set salaire=40000 where num_inf=256;
update infirmier set salaire=40000 where num_inf=256
       *
ERROR at line 1:
ORA-20006: Opération refusée: le nouveau salaire doit être supérieur à celui d
avant!
ORA-06512: at "DBAHOPITAL.MDF_INF", line 8
ORA-04088: error during execution of trigger 'DBAHOPITAL.MDF_INF'
```

Nous voyons que, l'insertion se fait normalement quand la modification augmente le salaire précédent. Dans le cas contraire le trigger empêche la modification et affiche le message d'erreur que nous avons personnalisé.

5) L'administrateur veut, pour un besoin interne, avoir le total des salaires des infirmiers pour chaque service. Pour cela, il ajoute un attribut : total_salaire_service dans la table service.
- Ajouter l'attribut.
- Créer un trigger TotalSalaire_Serivce_trigger qui met à jour l'attribut total_salaire_service après l'insertion d'un infirmier.
- Créer un trigger TotalSalaireUpdate_trigger qui met à jour l'attribut total_salaire_service après la mise à jour d'un salaire.

L'ajout de l'attribut «total_salaire_service» :

```
SQL> alter table service add total_salaire_service number(10) default 0;

Table altered.

SQL> desc service;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CODE_SERVICE                              NOT NULL VARCHAR2(3)
 NOM_SERVICE                                        VARCHAR2(50)
 BATIMENT                                           VARCHAR2(1)
 DIRECTEUR                                          NUMBER(3)
 TOTAL_SALAIRE_SERVICE                              NUMBER(10)
```

Code du trigger *TotalSalaire_Service_trigger :*

Nous ajoutons le salaire de l'infirmier inséré au total_salaire_service du service en entrée :

```
SQL> create or replace trigger TotalSalaire_Service_trigger after insert on infirmier f
or each row
  2  declare
  3  a number(10);
  4  n number(10);
  5  begin
  6  select total_salaire_service into a from service where code_service=:new.code_serv
ice;
  7  update service set total_salaire_service=a+:new.salaire where code_service=:new.co
de_service;
  8  select total_salaire_service into n from service where code_service=:new.code_serv
ice;
  9  dbms_output.put_line('L ancien total des salaires des infirmiers du service '||:ne
w.code_service||': '||a||'DA est devenu: '||n||'DA.');
 10  end;
 11  /

Trigger created.
```

Exemple d'execution :

```
SQL> delete from infirmier where num_inf=256;
Un infirmié supprimé!

1 row deleted.

SQL> select nom_service,total_salaire_service from service;

NOM_SERVICE                                        TOTAL_SALAIRE_SERVICE
-------------------------------------------------- ---------------------
Cardiologie                                                           0
Chirurgie generale                                                    0
Reanimation et Traumatologie                                          0

SQL> insert into infirmier values(255,'CHG','jour',40000);
L ancien total des salaires des infirmiers du service CHG: 0DA est devenu:
40000DA.
Un nouveau infirmié est affecté au CHG

1 row created.

SQL> insert into infirmier values(256,'CHG','jour',30000);
L ancien total des salaires des infirmiers du service CHG: 40000DA est devenu:
70000DA.
Un nouveau infirmié est affecté au CHG

1 row created.

SQL> select nom_service,total_salaire_service from service;

NOM_SERVICE                                        TOTAL_SALAIRE_SERVICE
-------------------------------------------------- ---------------------
Cardiologie                                                           0
Chirurgie generale                                                70000
Reanimation et Traumatologie                                          0
```

La valeur de total_salaire_service du service 'CHG' est effectivement modifiée (augmentée) après l'insertion d'un infirmier (ou plusieurs) affecté à ce service.

Code du trigger *TotalSalaireupdate_trigger :*

Puisque nous avons défini auparavant un trigger qui exige qu'une nouvelle valeur d'un salaire doit être supérieure à la précédente, nous n'avons pas à tester s'il s'agit de ce cas ou son contraire, mais nous ajoutons directement la différence entre le nouveau et l'ancien salaire au total_salaire_service du service concerné :

```
SQL> create or replace trigger TotalSalaireupdate_trigger after update of salaire on in
firmier for each row
  2  declare
  3  a number(10);
  4  n number(10);
  5  begin
  6  select total_salaire_service into a from service where code_service=:new.code_serv
ice;
  7  update service set total_salaire_service=a+:new.salaire-:old.salaire where code_se
rvice=:new.code_service;
  8  select total_salaire_service into n from service where code_service=:new.code_serv
ice;
  9  dbms_output.put_line('L ancien total des salaires des infirmiers du service '||:ne
w.code_service||': '||a||'DA est devenu: '||n||'DA.');
 10  end;
 11  /

Trigger created.
```

Exemple d'execution :

```
SQL> update infirmier set salaire=40000 where num_inf=256;
L infirmier n° 256,son ancien salaire est:30000DA et son nouveau salaire est:
40000DA.
L ancien total des salaires des infirmiers du service CHG: 70000DA est devenu:
80000DA.

1 row updated.

SQL> select nom_service,total_salaire_service from service;

NOM_SERVICE                                        TOTAL_SALAIRE_SERVICE
-------------------------------------------------- ---------------------
Cardiologie                                                            0
Chirurgie generale                                                 80000
Reanimation et Traumatologie                                           0
```

Nous voyons que le trigger modifie le total_salaire_service du service CHG après la modification du salaire de l'infirmier n° 256 qui appartient à ce service.

6) Un infirmier peut changer de service. Créer un trigger qui met à jour l'attribut total_salaire_service des deux services.

A chaque modification du code_service d'un infirmier, le salaire de ce dernier est soustrait de total_salaire_service de son service précédent et est additionné au total_salaire_service de son nouveau service :

```
SQL> create or replace trigger mdf_inf_srv before update of code_service on infirmier f
or each row
  2  declare
  3  n number(2);
  4  a number(10);
  5  nt number(10);
  6  e exception;
  7  begin
  8  select count(*) into n from service where code_service=:new.code_service;
  9  if(n=0) then raise e;
 10  else
 11  dbms_output.put_line('L infirmier n° '||:new.num_inf||',affecté au '||:old.code_se
rvice||' est reaffecté au '||:new.code_service||'!');
 12  select total_salaire_service into a from service where code_service=:old.code_serv
ice;
 13  update service set total_salaire_service=total_salaire_service-:new.salaire where
code_service=:old.code_service;
 14  select total_salaire_service into nt from service where code_service=:old.code_ser
vice;
 15  dbms_output.put_line('L ancien total des salaires des infirmiers du service '||:ol
d.code_service||': '||a||'DA est devenu: '||nt||'DA.');
 16  select total_salaire_service into a from service where code_service=:new.code_serv
ice;
 17  update service set total_salaire_service=total_salaire_service+:new.salaire where
code_service=:new.code_service;
 18  select total_salaire_service into nt from service where code_service=:new.code_ser
vice;
 19  dbms_output.put_line('L ancien total des salaires des infirmiers du service '||:ne
w.code_service||': '||a||'DA est devenu: '||nt||'DA.');
 20  end if;
 21  EXCEPTION
 22  when e then raise_application_error(-20006,'Opération refusée: code de service ent
ré invalide!');
 23  end;
 24  /

Trigger created.
```

Exemple d'execution :

```
SQL> select code_service,total_salaire_service from service;

COD TOTAL_SALAIRE_SERVICE
--- --------------------
CAR                    0
CHG                80000
REA                    0

SQL> update infirmier set code_service='CAR' where num_inf=256;
L infirmier n° 256,affecté au CHG est reaffecté au CAR!
L ancien total des salaires des infirmiers du service CHG: 80000DA est devenu:
40000DA.
L ancien total des salaires des infirmiers du service CAR: 0DA est devenu:
40000DA.

1 row updated.

SQL> select code_service,total_salaire_service from service;

COD TOTAL_SALAIRE_SERVICE
--- --------------------
CAR                40000
CHG                40000
REA                    0
```

Après affectation de l'infirmier n°256 au CAR, précédemment au CHG, les valeurs des total_salaire_service des deux services sont modifiées (respectivement : augmentée, diminuée).

7) L'administrateur veut sauvegarder toutes les hospitalisations des patients dans le temps. A chaque fois qu'un patient est hospitalisé, une ligne sur les informations de son hospitalisation est sauvegardée dans une autre table « Hist_Hospit ». La table « Hist_Hospit » est définie par Hist_Hospit (date_hospit,num_patient code_service*). Où date_hospit est la date d'hospitalisation.

Création de la table *Hist_Hospit :*

```
SQL> desc hospitalisation
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 NUM_PATIENT                               NOT NULL NUMBER(3)
 CODE_SERVICE                                       VARCHAR2(3)
 NUM_CHAMBRE                                        NUMBER(3)
 LIT                                                NUMBER(1)
 DATE_HOST                                          DATE

SQL> create table hist_hospit(date_hospit date,num_patient number(3),code_service varchar2(3),
constraint hist_hospitpk primary key (date_hospit,num_patient),constraint hist_hospitfk1 forei
gn key (num_patient) references hospitalisation,constraint hist_hospitfk2 foreign key (code_se
rvice) references service);

Table created.

SQL> desc hist_hospit;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 DATE_HOSPIT                               NOT NULL DATE
 NUM_PATIENT                               NOT NULL NUMBER(3)
 CODE_SERVICE                                       VARCHAR2(3)
```

Création du trigger qui remplit la table *Hist_Hospit :*

```
SQL> create or replace trigger hist_hosp after insert on hospitalisation for each row
  2  begin
  3  insert into hist_hospit values(:new.date_host,:new.num_patient,:new.code_service);
  4  end;
  5  /

Trigger created.
```

Exemple d'execution :

Nous choisissons par exemple le service REA et nous cherchons des lits libres pour hospitaliser le patient :

```
SQL> select h.num_chambre,c.nb_lits,h.lit as Lits_occupes from hospitalisation h,cham
bre c where h.code_service='REA' and h.code_service=c.code_service and c.num_chambre=
h.num_chambre group by h.num_chambre,c.nb_lits,h.lit order by h.num_chambre;

NUM_CHAMBRE    NB_LITS LITS_OCCUPES
----------- ---------- ------------
        101          1            1
        102          1            1
        103          2            1
        103          2            2
        104          2            1
        104          2            2
        105          1            1
        107          2            1
        107          2            2
        108          2            1

10 rows selected.

SQL> insert into hospitalisation values(13,'REA',108,2,'02/11/2018');

1 row created.

SQL> select * from hist_hospit;

DATE_HOS NUM_PATIENT COD
-------- ----------- ---
02/11/18          13 REA
```

Les informations de l'hospitalisation du patient n°13 sont sauvegardées dans la table Hist_Hospit.