Cartouche du document

Année : ING 2
Activité : Travail dirigé

Objectifs

Revoir le langage SQL étudié en première année.

Appliquer des notions simples de PL/SQL à travers des triggers

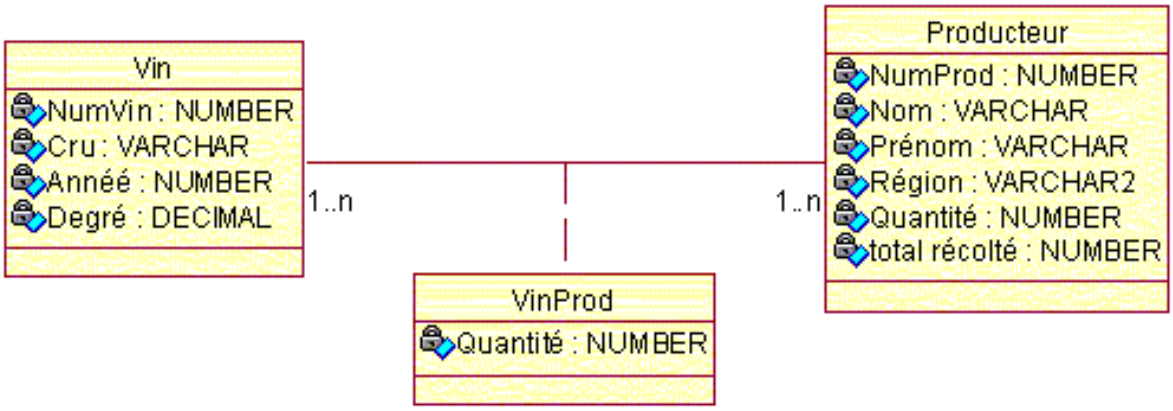
Sommaire des exercices

Corps des exercices

1 - Le producteur de vin

Énoncé :

Dans cet exercice, on travaille avec le modèle conceptuel de données suivant :



Question 1)

Énoncé de la question

Etablir le MLD correspondant

Solution de la question

VIN (numvin,cru,année,degré)

PRODUCTEUR (numprod,nom,prenom,région,quantité total récolté)

Association :

RECOLTE : (#numvin,#numprod,quantité)

Question 2)

Énoncé de la question

Etablir le script de création des tables

Solution de la question

Vous pouvez consulter le code SQL du

```
/*****************************************/
/*****PREMIERE PARTIE  ******************/
/*****************************************/
-- Création des tables

drop table recolte;
drop table vin;
drop table producteur;

create table vin
(numvin varchar(5) constraint pk_vin primary key,
cru varchar(10),
annee number(4),
degre number(4,1)
);

create table producteur
(numprod varchar(5) constraint pk_prod primary key,
nom varchar(30),
prenom varchar(30),
region varchar(30),
qte_tot_rec number(6,2)
```

```
);

create table recolte
(numvin  varchar(5),
numprod varchar(5),
quantite number(6,2),
constraint pk_recolte primary key(numvin,numprod)
);

-- script de création des clés étrangères

alter table recolte add constraint fk_recolte1 foreign key(numvin)
references vin(numvin);
alter table recolte add constraint fk_recolte2 foreign key(numprod)
references producteur(numprod);
```

.

Question 3)

Énoncé de la question

Trouver quelques données en vue de la question suivante

Solution de la question

Vous pouvez consulter le code SQL du

```
-- jeu d essai de départ



insert into vin values ('v1','c1',1994,12);

insert into vin values ('v2','c1',1994,13);
```

```
insert into vin values ('v3','c2',1992,10);



insert into producteur values ('p1','Rotchild','','Beaujolais',0);

insert into producteur values ('p2','Toto','','Jura',0);

insert into producteur values ('p3','titi','','Beaujolais',0);

insert into producteur values ('p4','tata','','Jura',0);
```

.

Question 4)

Énoncé de la question

Pour les tables Producteur et Vin ecrire pour chacune un trigger qui permet de générer automatiquement les clés (utilisation d'une séquence).

Solution de la question

Question 5)

Énoncé de la question

Ecrire en PL/SQL un déclencheur (trigger) permettant de mettre à jour l'attribut Quantité totale à chaque insertion d'une ligne dans la table récolte.

Solution de la question

Réponse globale à tout l'exercice

Vous pouvez consulter le code SQL du

```
-- script de création du trigger



CREATE OR REPLACE TRIGGER majprod
```

```
BEFORE insert on recolte for each row

begin

update producteur set qte_tot_rec=qte_tot_rec + :new.quantite

where numprod = :new.numprod ;

end;

/



-- jeu d essai en vue de tester le trigger



insert into recolte values ('v1','p1',100);

insert into recolte values ('v2','p1',200);

insert into recolte values ('v2','p4',100);



select numprod,qte_tot_rec from producteur ;



/*****************************************

NUMPR QTE_TOT_REC

----- -----------
```

```
p1          300

p2           0

p3           0

p4          100

*********************************************/
```

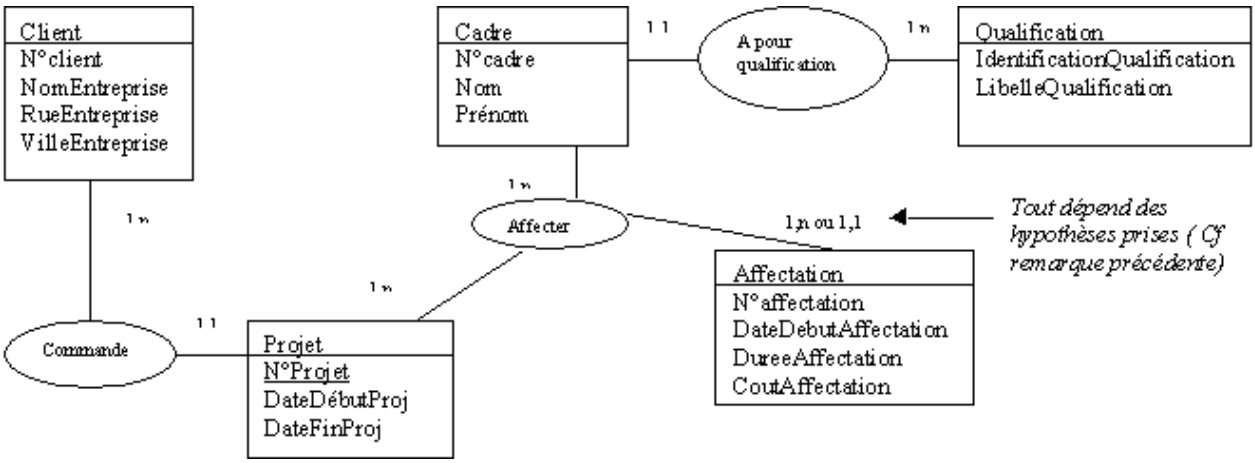
.

2 - SSII

Énoncé :

Soit une société de services souhaitant gérer l'affectation de ces cadres en clientèle, sachant que :

ce qui donne le MCD (modèle conceptuel de données) correspondant :



Question 1)

Énoncé de la question

PL/SQL : Si on désirait ajouter à la table Client un attribut NombreProjet qui permet de savoir à tout instant combien de projets a commandé un client, quel outil SQL faudrait-il mettre en place pour mettre automatiquement à jour cette information ?

Solution de la question

Il s'agit bien sûr d'écrire un trigger qui permet de mettre à jour automatiquement la table client dès qu'un nouveau projet est créé.

Question 2)

Énoncé de la question

On vous demande de réaliser cet objet.

Solution de la question

Vous pouvez consulter le code SQL du

```
/*********************************************/

/*******DEUXIEME PARTIE ********************/

/*********************************************/



-- script de création des tables




/*************************************************************/

-- création du Trigger

-- on a besoin de la table client et de la table projet pour pouvoir tester le trigger




drop table projet cascade constraint;
```

```
drop table clientp cascade constraint ;



CREATE TABLE CLIENTP

(

NoClient number(5) PRIMARY KEY,

nbprojet number(5)

);



CREATE TABLE projet

(

NoProjet number(5) PRIMARY KEY,

DateDebutProj date,

DateFinProj date,

NoClient number(5)

) ;



ALTER TABLE projet

ADD CONSTRAINT FK_projet
```

```
FOREIGN KEY (NoClient) REFERENCES clientp(NoClient) ;




/*********************************/

/* creation du triggger en insert*/

/*********************************/

CREATE OR REPLACE TRIGGER maj_cli

BEFORE INSERT ON projet FOR EACH ROW

BEGIN

UPDATE clientp set nbprojet=nbprojet+1 where NoClient =:New.NoClient ;

EXCEPTION

WHEN NO_DATA_FOUND THEN

  INSERT INTO CLIENTp (Noclient,NbProjet) VALUES (:NEW.NoClient,1);

END ;

/



-- TEST du TRIGGER --

-- On test d abord quand un client existe

insert into clientp values (1,0);
```

INSERT INTO projet (Noprojet,Noclient) values (1,1);

/*******************************

SQL> select * from projet;

NOPROJET	DATEDEBUT	DATEFINPR	NOCLIENT
1			1

SQL> select * from clientp;

NOCLIENT	NBPROJET
1	1

***********************************/

-- On test maintenant quand un client n existe pas

INSERT INTO projet (Noprojet,Noclient) values (2,2);

/**

On obtient une erreur :

INSERT INTO projet (Noprojet,Noclient) values (2,2);

*

ERROR at line 1:

ORA-02291: integrity constraint (JB.FK_PROJET) violated - parent key not found

En effet il existe une contrainte de clé étrangère entre clientp et projet

Il Faut donc vérifier que le client existe pour vérifier la contrainte

de clé étrangère avant oracle

alorson rajoute un select avant update alorson vérifie ds client que

le client existe : si il existe alors OK si non on passe ds la gestion

de l erreur et on insert le client avt l insertion de projet

**/

```
CREATE OR REPLACE TRIGGER maj_cli

BEFORE INSERT ON projet FOR EACH ROW

DECLARE

-- declaration de vnoclient du même type que la col. noclient de

-- la table clientp



VNoclient Clientp.Noclient%TYPE ;

BEGIN

-- on test si le nuémro client existe

SELECT Noclient INTO Vnoclient from clientp where Noclient=:NEW.NoClient;

-- si inexistant alors génération de l'exception WHEN NO_DATA_FOUND

-- si existant on met à jour le nb de projet du client

UPDATE clientp set nbprojet=nbprojet+1 where NoClient =:New.NoClient ;

EXCEPTION

WHEN NO_DATA_FOUND THEN

  INSERT INTO CLIENTp (Noclient,NbProjet) VALUES (:NEW.NoClient,1);

END ;

/
```

INSERT INTO projet (Noprojet,Noclient) values (2,2);

-- 1 row created.

/***

SQL> select * from clientp;

NOCLIENT	NBPROJET
1	1
2	1

SQL> select * from projet;

NOPROJET	DATEDEBUT	DATEFINPR	NOCLIENT
1			1

```
      2                    2

**********************************************************************/



/**************************************/

/* creation du triggger en suppression*/

/**************************************/

CREATE OR REPLACE TRIGGER maj_cli

BEFORE DELETE ON projet FOR EACH ROW

BEGIN

UPDATE clientp set nbprojet=nbprojet-1 where NoClient =:old.NoClient ;

END ;

/



/*********************************************

SQL> delete from projet where NoProjet=2;



1 row deleted.
```

```
SQL> select * from clientp;



 NOCLIENT  NBPROJET

--------- ---------

       1         1

       2         0

**********************************************/



/*************************************/

/* creation du triggger en mise à jour*/

/*************************************/

CREATE OR REPLACE TRIGGER maj_cli

BEFORE UPDATE ON projet FOR EACH ROW

DECLARE

VNoclient Clientp.Noclient%TYPE ;

BEGIN

-- Ds tous les cas on enlève un projet à l ancien client

UPDATE clientp set nbprojet=nbprojet-1 where NoClient =:old.NoClient ;

-- on test si le nuémro client existe
```

```
select noclient into Vnoclient from clientp where noclient=:new.noclient;

-- si inexistant alors génération de l'exception WHEN NO_DATA_FOUND

-- si existant on met à jour le nb de projet du client

update clientp set nbprojet=nbprojet+1 where Noclient=:new.noclient ;

EXCEPTION

WHEN NO_DATA_FOUND THEN

-- on insert le nouveau client en mettant son nbprojet à 1

  INSERT INTO CLIENTp (Noclient,NbProjet) VALUES (:NEW.NoClient,1);

END ;

/
```

/**********

Supposons l état de nos tables suivant :

PROJET

NOPROJET DATEDEBUT DATEFINPR NOCLIENT

-------- --------- --------- ---------

1 1

2 1

CLIENTP

NOCLIENT NBPROJET

--------- ---------

1 2

2 0

**********/

-- On met à jour la table projet on affecte le projet 2 au client 2 et non au client 1

UPDATE projet set noclient=2 where noprojet = 2 ;

/**********

```
SQL> select * from clientp;

 NOCLIENT  NBPROJET
--------- ---------
       1         1
       2         1

SQL> select * from projet ;

 NOPROJET DATEDEBUT DATEFINPR  NOCLIENT
--------- --------- --------- ---------
       1                           1
       2                           2

**********************************************************/

-- On met à jour la table projet on affecte le projet 2 au client 3 et non au client 2
-- ATTENTION Client 3 inexistant

UPDATE projet set noclient=3 where noprojet = 2 ;
```

On obtient :

/***

SQL> select * from clientp;

NOCLIENT NBPROJET

--------- ---------

1 1

2 0

3 1

SQL> select * from projet ;

NOPROJET DATEDEBUT DATEFINPR NOCLIENT

--------- --------- --------- ---------

1 1

2 3

***/

-- Remarque : il est possible de regrouper les trois actions ds un seul trigger

```
CREATE OR REPLACE TRIGGER maj_cli

BEFORE INSERT OR UPDATE OR DELETE ON projet FOR EACH ROW

DECLARE

VNoclient Clientp.Noclient%TYPE ;

BEGIN

IF DELETING THEN

  UPDATE clientp set nbprojet=nbprojet-1 where NoClient =:old.NoClient ;

ELSIF INSERTING  THEN

  select noclient into Vnoclient from clientp where noclient=:new.noclient;

  UPDATE clientp set nbprojet=nbprojet+1 where NoClient =:New.NoClient ;

ELSIF UPDATING  THEN

  UPDATE clientp set nbprojet=nbprojet-1 where NoClient =:old.NoClient ;

  select noclient into Vnoclient from clientp where noclient=:new.noclient;
```

```
  update clientp set nbprojet=nbprojet+1 where Noclient=:new.noclient ;

END IF;



EXCEPTION

WHEN NO_DATA_FOUND THEN

  INSERT INTO CLIENTp (Noclient,NbProjet) VALUES (:NEW.NoClient,1);



END ;

/
```

.