

Solution TD3 : Le modèle Objet relationnel

Exercice :

Schéma Objet relationnel :

```
Type T_Produit : <num: integer, designation: string, prix: float, stock: integer>
Produit de T_Produit (#num)
Type T_telephone tableau fixe(10) de string
Type T_Adresse : <num:integer, rue:string, ville:string, codepost:char(5)>
Type T_Client : <num:integer, nom:string, prenom:string, adresse:T_Adresse, date_nais:date,
                tel:T_telephone, sexe:char={'m', 'f'}>
Client de T_Client(#num)
Type T_LigneFacture : <Numligne :integer, fkProduit =>Produit, qte:integer>
Type T_LignesFacture : collection de <T_LigneFacture>
Type T_Facture : <numfact:integer, datefact:date, dateexp:date, adresse : T_Adresse,
                Ligfacture:T_LignesFacture, fkClient =>Client, total():float>
Facture de T_Facture(#num)
```

1- Création de la base de données

1. Réalisez l'implémentation SQL3 sous Oracle.

Implémentation sous Oracle

```
-- Type et table pour Produit
CREATE TYPE T_Produit AS OBJECT ( Prodnum NUMBER, Designation VARCHAR(20), Prix NUMBER,
                                Stock NUMBER );
/
CREATE TABLE Produit OF T_Produit (PRIMARY KEY (Prodnum) );
/
-- Types et table pour Client
CREATE TYPE T_Adresse AS OBJECT ( Num NUMBER, Rue VARCHAR(20), Ville VARCHAR(20), Codepost
                                CHAR(5) );
/
CREATE TYPE T_telephone AS VARRAY(10) OF VARCHAR(20);
/
CREATE TYPE T_Client AS OBJECT ( num NUMBER, nom VARCHAR(20), prenom VARCHAR(20),
                                adresse T_Adresse, date_nais DATE, tel T_telephone, sexe
                                CHAR(1));
/
CREATE TABLE Client OF T_Client (PRIMARY KEY (num), CHECK (sexe IN ('m', 'f')));
/
-- Type ligne de facture et collection de ligne de facture
CREATE TYPE T_LigneFacture AS OBJECT ( Numligne NUMBER, fkProduit REF T_Produit,
                                       qte NUMBER);
/
```

```
CREATE TYPE T_LignesFacture AS TABLE OF T_LigneFacture;
/
-- Type et table pour Facture
CREATE TYPE T_Facture AS OBJECT ( numfact Number, date_etabli DATE, date_exp DATE,
                                adr_dest T_adresse, ligfacture T_LignesFacture,
                                fkClient REF T_Client, MEMBER FUNCTION total RETURN
                                NUMBER );
/
CREATE TABLE Facture OF T_Facture ( PRIMARY KEY (numfact), SCOPE FOR (fkClient) IS Client )
NESTED TABLE ligfacture STORE AS nt_facture_lignes_facture;
%SCOPE FOR Limite la portée de la référence à une table particulière
```

2. Réalisez l'implémentation de la méthode total()

```
CREATE OR REPLACE TYPE BODY T_Facture AS
MEMBER FUNCTION total RETURN NUMBER IS
    vTotal NUMBER;
BEGIN
    SELECT SUM(lf.qte*lf.fkProduit.prix) INTO vTotal
    FROM TABLE(SELECT ligfacture FROM facture f WHERE f.numfact=SELF.numfact)lf;
    RETURN vTotal;
END total;
END;
```

3. Initialiser les tables client et produit avec les données de votre choix (au moins deux clients et deux produits).

Insertion de deux clients

```
INSERT INTO client VALUES (T_Client(1, 'Smaili', 'Mohamed', T_Adresse(102, '5 juillet', 'Annaba',
23000), '06-15-1970', T_telephone('0551558124'), 'm'));
INSERT INTO client VALUES (2, 'Kadri', 'Ali', t_adresse(07, '5 juillet', 'Annaba', 23000), '03-20-1985',
T_telephone('0559999999'), 'm');
```

Insertion de quelques produits

```
INSERT INTO Produit VALUES (1, 'Imprimante', 8500, 100);
INSERT INTO Produit VALUES (2, 'Ecran', 8000, 20);
INSERT INTO Produit VALUES (3, 'Clavier', 500, 50);
```

4. Initialiser la BD avec les données de votre choix (deux factures de deux lignes chacune au moins).

```
DECLARE
client_1 REF T_Client;
client_2 REF T_Client;
produit_1 REF T_Produit;
produit_2 REF T_Produit;
produit_3 REF T_Produit;
BEGIN
SELECT REF(c) INTO client_1 FROM Client c WHERE c.num=1;
SELECT REF(c) INTO client_2 FROM Client c WHERE c.num=2;
SELECT REF(p) INTO produit_1 FROM Produit p WHERE p.prodnum=1;
SELECT REF(p) INTO produit_2 FROM Produit p WHERE p.prodnum=2;
SELECT REF(p) INTO produit_3 FROM Produit p WHERE p.prodnum=3;
INSERT INTO Facture VALUES (1, '01-12-2015','02-12-2015',T_Adresse(102, '5 juillet', 'annaba',
23000), T_LignesFacture (T_LigneFacture (1, produit_1, 10),
T_LigneFacture(2,produit_2,5)), client_1);
INSERT INTO Facture VALUES ( 2, '02-12-2015','03-12-2015', T_Adresse (07, '5 juillet', 'Souk-ahras',
41000), T_LignesFacture(T_LigneFacture (1, produit_1, 15),
T_LigneFacture(2,produit_3,7)), client_2);
INSERT INTO Facture VALUES ( 3, '01-12-2015','02-12-2015', T_Adresse (07, '5 juillet', 'Souk-ahras',
41000), T_LignesFacture (T_LigneFacture (1, produit_1, 5),
T_LigneFacture (2,produit_2,8), T_LigneFacture (3, produit_3, 7)),
client_2);
END;
```

2- Interrogation de la base de données

5. Ecrire une requête SQL permettant d'afficher la liste des factures existantes ordonnées par leurs numéros.

```
SELECT * FROM facture F
ORDER BY F.numfact ;
/ ou
SELECT F.numfact, F.date_etabli, F.date_exp , F.adr_dest.num, F.adr_dest.rue, F.adr_dest.ville,
F.adr_dest.codepost FROM facture F
ORDER BY F.numfact ;
```

6. Ecrire une requête SQL permettant d'afficher la liste des factures existantes, avec le nom et le prénom du client.

```
SELECT f.numfact AS numero, f.fkClient.nom AS nom, f.fkClient.prenom AS prenom
FROM Facture f;
/
```

7. Ecrire une requête SQL permettant d'afficher le montant total de chaque facture ainsi que le nom et le prénom du client.

```
SELECT f.numfact AS numero, f.fkClient.nom AS nom, f.fkClient.prenom AS prenom, f.total() AS total
FROM Facture f;
/
```

8. Ecrire une requête SQL permettant d'afficher le numéro de la facture, le nom du client et la désignation des produits commandés.

```
SELECT f.numfact, f.fkclient.nom,
CURSOR ( SELECT L.fkproduit.designation
          FROM TABLE (f.ligfacture) L)
FROM Facture f;
/
% Pour pouvoir récupérer les informations de plusieurs tables imbriquées simultanément, on utilise la
commande CURSOR
/ou
/
SELECT f.numfact, f.fkclient.nom, L.fkproduit.designation FROM facture f, (TABLE(f.ligfacture) L)
/
```

9. Ecrire une requête SQL permettant d'afficher les numéros des factures et les noms des clients qui ont commandé le produit 2.

```
SELECT f.numfact, f.fkclient.nom
FROM Facture f, TABLE (f.Ligfacture) L
WHERE L.fkproduit.prodnum = 2 ;
/
```

10. Écrire une requête SQL permettant de renvoyer le montant total de toutes les factures, ainsi que le nombre de factures, du client 2.

```
SELECT COUNT(*) as nombre, SUM(f.total()) AS montant
FROM Facture f
WHERE f.fkClient.num = 2;
/
```

11. Écrire une requête SQL permettant d'afficher le numéro de facture et le nombre de produit commandé pour chaque facture.

```
SELECT f.numfact, CURSOR ( SELECT COUNT(L.Numligne)AS nbr_produit
                          FROM TABLE(f.LigFacture) L)
FROM facture f;
/
```

12. Écrire une requête SQL permettant de renvoyer le montant moyen des factures du client nommé Kadri.

```
SELECT AVG(f.total()) AS montant
FROM Facture f
WHERE f.fkClient.nom = 'Kadri';
/
```

13. Écrire une requête SQL permettant de renvoyer les numéros et les noms des clients ayant payé au moins une facture supérieure à 10000 DA.

```
SELECT DISTINCT f.fkClient.num AS numero, f.fkClient.nom AS nom
FROM Facture f
WHERE f.total() > 10000;
/
```

14. Écrire une requête SQL permettant de renvoyer le montant total, le montant moyen et le nombre des factures pour chaque client.

```
SELECT SUM(f.total()) AS total, AVG (f.total()) AS moyen, COUNT(f.numfact) AS nombre
FROM Facture f
GROUP BY f.fkClient; /
```

15. Écrire une requête SQL permettant de renvoyer les numéros et noms des clients ayant payé au moins une facture avec plus de 3 produits.

```
SELECT DISTINCT f.fkClient.num AS numero, f.fkClient.nom AS nom
FROM Facture f
WHERE ( SELECT COUNT(*) FROM TABLE(f.Ligfacture))>3;
/
```

3- Mise à jour

16. Ajouter une ligne de facturation dans la facture 1.

```
DECLARE
produit_1 REF T_Produit;
BEGIN
SELECT REF(p) INTO produit_1
FROM Produit p
WHERE p.prodnum=1;
INSERT INTO TABLE (SELECT f.ligfacture
FROM Facture f
WHERE f.numfact = 1)
VALUES (T_LigneFacture(3,produit_1,75));

END;
/
```

17. supprimer la ligne de facturation 2 de la facture 2.

```
DELETE FROM TABLE(SELECT f.ligfacture
                      FROM Facture f
                      WHERE f.numfact=2) L
WHERE L.numLigne=2
/
```

18. Ajouter un nouveau numéro de téléphone pour le client 3.

```
DECLARE
tableau_tel T_telephone;
BEGIN
  SELECT tel INTO tableau_tel FROM Client c WHERE c.num = 2 ;
  tableau_tel.EXTEND;
  tableau_tel(tableau_tel.LAST) := ('555555555555');
  UPDATE Client SET tel = tableau_tel WHERE num = 3;
END;
/
```

19. Modifier la ligne de facturation 2 en sélectionnant le produit 3.

```
UPDATE TABLE (SELECT f.ligfacture
                 FROM Facture f
                 WHERE f.numfact=1) L
SET L.fkproduit = (SELECT REF(p)
                  FROM Produit p
                  WHERE p.prodnum=3)
WHERE L.numligne=2;
```