Chapitre 5 : PL/SQL : Premières notions

5.1 - Introduction

• PL/SQL : Langage de programmation
- utilisant les commandes SQL
- disposant d'un ensemble de fonctionnalités supplémentaires (programmation structurée)
• Déclarations
variables
variables initialisées
constantes
• Saisie au clavier par &valeur
• Structures algorithmiques (itérations, alternatives)

5.2 - Structure d'un bloc PL/SQL

Up

5.2.1 Schéma d'un script intégrant un bloc PL/SQL

```
- - bloc de commandes SQL

        _____ ;
        _____ ;
- - fin commandes SQL

- - Bloc PL/SQL

        DECLARE
            - - déclaration de variables et de curseurs
            _____ ;
            _____ ;
        BEGIN
            - - bloc d'instructions
            _____ ;
            _____ ;
        [EXCEPTION
           traitement des erreurs ]

        END;

- - fin du bloc pl/sql
/    - - exécution du bloc PL/SQL ; sans espace en début de ligne

- - autres commandes SQL pour sortir les résultats
```

Remarque : on peut créer une table temporaire en SQL, on l'utilise en PL/SQL pour y stocker des résultats, puis en SQL, à nouveau, on affiche les résultats et on détruit la table temporaire.
Exemple :

```
-- nombre d'artistes nés après une année donnée en paramètre
CREATE TABLE temp
  (nombre NUMBER(3));
DECLARE
  ANNEE NUMBER(4);
  Xnombre NUMBER(3);
BEGIN
  ANNEE:=&annee_naissance;
  SELECT COUNT(*) INTO Xnombre
   FROM A_artiste
   WHERE datns > ANNEE;
  INSERT INTO temp VALUES(Xnombre);
  COMMIT;
END;
/
SELECT * FROM temp;
```

5.2.2 Détails du script PL/SQL

PARTIE DECLARATION :

a) Variables

DECLARE nom_variable TYPE;

cas particuliers de types :

nom_variable nom_table.nom_colonne%TYPE ; (variable de même type que la colonne)

nom_variable nom_table%ROWTYPE ; (record avec les mêmes champs que la table)

variable initialisée

pays VARCHAR2(20):='FRANCE' ;

b) Constantes

DECLARE nom_constante CONSTANT TYPE;

pays CONSTANT VARCHAR2(20):='FRANCE' ;

La valeur de la constante ne peut pas être modifiée dans le bloc PlSql.

c) Curseurs (vus au chapître suivant)

```
DECLARE CURSOR nom_curseur IS
  SELECT nom_colonne(s)
   FROM nom_table(s)
   WHERE condition
   [ORDER BY nom_colonne(s)];
```

CORPS DU BLOC :

Bloc

Le corps d'un bloc PlSql est tout entier enfermé entre BEGIN et END;

S'il est vide, le mot BEGIN est omis mais END; demeure.

```
  [BEGIN ...] END;
```

Saisie de données au clavier avec &nom

Variables internes
   nom_variable TYPE;
   nom_variable TYPE := valeur_initiale;
Affectation
   :=
Insertion
   INSERT INTO nom_table VALUES (var_curseur.nom_colonnes(s))
Mise à jour d'une table :

```
DECLARE
BEGIN
  UPDATE f_agenda
    SET syn='francette',location = 'monet.org'
    WHERE syn = 'francette.monet';
 COMMIT;
END;
/
PROMPT 'Après mise à jour : ';
SELECT nom,syn||'@'||location
  FROM f_agenda ;

NOM
------------------------------
SYN||'@'||LOCATION
--------------------------------------------------
Francette Monet
fm@info.univ-angers.fr

Francette Monet
francette@monet.org
```

Ouverture, utilisation et fermeture d'un curseur
  OPEN nom_curseur... ;
  FETCH nom_curseur INTO nom_variable(s) ;
  UPDATE nom_table ... WHERE CURRENT OF CURSOR;
  CLOSE nom_curseur ;

5.3 - Structures algorithmiques Up

a) Boucle LOOP - END LOOP *"empaquetage des instructions"*

LOOP
  Bloc d'instructions avec

```
        EXIT WHEN condition;
      END LOOP;
```
Boucle FOR
```
      FOR condition -- l'indice de boucle n'est pas déclaré
        LOOP
          Bloc d'instructions;
        END LOOP;
```
Boucle WHILE
```
      WHILE condition
        LOOP
          Bloc d'instructions;
        END LOOP;
```
Alternatives/conditionnelles
```
      IF condition
        THEN Bloc_Instructions
        [ELSE
          Bloc_Instructions]
      END IF;
```
Alternatives emboîtées
```
      IF condition
        THEN Bloc_Instructions
        ELSIF condition
          THEN Bloc_Instructions
          ELSE Bloc_Instructions
      END IF;
```
Action nulle
```
        NULL ;
```
Séquence
```
      GOTO nom ;
      Bloc_Instructions ;
      ----------
      ----------
      <<nom>>
        Bloc_Instructions ;
```

5.4 - Saisie de données au clavier Up

&nom : saisie d'un mot au clavier ; un mot est un nombre ou une chaîne de caractères alphanumériques (éventuellement entre apostrophes)
Oracle substitura la suite des caractères saisis à &nom.
'&nom' ou '%&nom%' : permettront de ne saisir que les caractères indispensables.

Pour marquer le fin du paramètre de saisie, on peut mettre un point.

```
SELECT prénom, nom, telephone
FROM CLIENT
WHERE nom LIKE '%M&chaine.ET%';
```

5.5 - Affichages de résultats Up

On utilise le package (ensemble de procédures et fonctions) **DBMS_OUTPUT**.
Tout d'abord, avant le bloc PL/SQL, utiliser l'instruction :

set server output on

Puis pour chaque affichage :

dbms_output.put_line('texte'||X);

Nous remarquons la concaténation de chaînes de caractères avec : **||**

5.6 - Exemple

Exemple de bloc PL/SQL

Il s'agit d'automatiser une séquence d'Instructions.

A partir du code_client, déterminer sa dernière commande et calculer le montant de sa facture.

```
 SET SERVER OUTPUT ON

DECLARE
  xclient client%rowtype;
  xnum client.code_client%type;
  xnb NUMBER;
  xcom commande.no_commande%type;
  xmontant NUMBER(10,2);

BEGIN
  -- recherche du client
  xnum := '&code_du_client';
  SELECT * INTO xclient
    FROM client
    WHERE code_client = xnum;
  -- dernière commande
  SELECT COUNT(*) INTO xnb
    FROM commande
    WHERE code_client = xnum;
  IF xnb = 0
    THEN
```

```
    DBMS_OUTPUT.PUT_LINE(xclient.prenom||' '||xclient.nom||' n''a pas de
commande.');
  ELSE
    DBMS_OUTPUT.PUT_LINE(xclient.prenom||' '||xclient.nom);
    DBMS_OUTPUT.PUT_LINE(xclient.adresse||' '||xclient.code_postal||'
'||xclient.ville);
    -- dernière commande
    SELECT no_commande INTO xcom
      FROM commande
      WHERE code_client = xnum
      AND date_commande = (SELECT MAX(date_commande)
                           FROM commande
                           WHERE code_client = xnum);
    -- calcul du montant de la dernière facture
    SELECT SUM(prix*qte) INTO xmontant
      FROM article_commande ac, catalogue c
      WHERE ac.reference = c.reference
      AND ac.no_commande = xcom;
    DBMS_OUTPUT.PUT_LINE('Montant de la facture : '||xmontant||' F');
  ENDIF;
END;
/
```

5.7 - Exercices

FEUILLE D'EXERCICES SQL, SQL*PLUS N°05 : PLsql

1. BLOC PLsql, VALEURS SAISIES AU CLAVIER, ALTERNATIVE.
Considérons les artistes de deux pays choisis par
l'utilisateur.
Déterminons le nombre d'oeuvres et la moyenne des valeurs de ces oeuvres
pour chacun des deux pays.
Affichons les résultats : NB_OEUVRES VALEUR_MOYENNE précédés d'une phrase
dépendant du résultat, par exemple : les artistes du pays 1 vendent plus
cher que les artistes du pays 2, ou le contraire.

2. BLOC PLSQL - AFFICHAGE DES RESULTATS

Déterminer le nombre d'artistes et le nombre moyen d'oeuvres par artiste.

"En moyenne, les xxx artistes ont réalisé chacun yyy oeuvres."

3. BLOC PLSQL

Un nouvel artiste vient faire une exposition dans un musée répertorié.

- Quelles sont les tables concernées par cette action ?
- Créer et remplir ces tables à partir des tables fm.A_xxx.
- Ecrire un algorithme permettant d'insérer les nouvelles données dans les tables.
- Les nouvelles données sont saisies au clavier.
- Ces saisies et les mises à jour des tables sont faites dans un bloc PLSQL.
- Vérifier par des requêtes que les nouvelles données sont bien là.

3. BLOC PLSQL - CURSEUR - BOUCLE FOR...

Sélectionner les n premières oeuvres par ordre décroissant des
valeurs.

Remarque : les valeurs existent nécessairement.

code_oeuvre titre valeur

4. BLOC PLSQL - CURSEUR - BOUCLE LOOP ... END LOOP

Artistes ayant le plus grand nombre d'oeuvres répertoriées

nom date_naissance localite pays nb_oeuvres

5. Liste concaténée des artistes par musée, ayant exposé entre
deux dates données.

Chapitre 6 : PL/SQL - Curseurs

6.1 - Introduction

Une requête retourne un ensemble d'occurrences indissociables.
Il se peut que nous ayons besoin de traiter une partie seulement de
ces occurrences ou de ne retenir que les n premières occurrences trouvées.
Par exemple, trouver les trois meilleures commandes, modifier certaines des occurrences obtenues.

6.2 - Définition d'un curseur sans paramètre

Up

- Déclarations liées au curseur
- DECLARE
- -- déclaration et définition du curseur lui-même
- nom_curseur CURSOR IS
- SELECT ... ;
- -- déclarations des variables utiles pour l'utilisation du curseur,
- -- par exemple une variable LIGNE pour récupérer une occurrence du curseur.
- -- Il s'agit d'une variable de type enregistrement avec pour champs, les colonnes
- -- obtenues dans le curseur.
- LIGNE nom_curs%ROWTYPE ;
- Utilisation du curseur
- -- ouverture du curseur
- OPEN nom_curs ;
- *C'est à l'ouverture que la sélection est exécutée*
-
- -- Lecture de la ligne courante du curseur et stockage dans la variable ligne
- FETCH nom_curs INTO ligne ;
- -- Accès aux champs (ou colonnes) par ligne.nom_champ
- --

6.4 exercice

TPsql_5 : PLsql - Curseurs

1. BLOC PLsql, VALEURS SAISIES AU CLAVIER, ALTERNATIVE.
Considérons les artistes de deux pays choisis par
l'utilisateur.
Déterminons le nombre d'oeuvres et la moyenne des valeurs des
artistes pour chacun des deux pays.
Affichons par exemple que les artistes du pays 1 vendent plus
cher que les artistes du pays 2, ou le contraire.

2. BLOC PLSQL - AFFICHAGE DES RESULTATS
Déterminer le nombre d'artistes et le nombre moyen d'oeuvres par artiste.

"Les xxx artistes ont réalisé en moyenne yyy oeuvres."

3. BLOC PLSQL
Un nouvel artiste vient faire une exposition dans un musée répertorié.
- Quelles sont les tables concernées par cette action ?
- Créer et remplir ces tables à partir des tables fm.A_xxx.
- Ecrire un algorithme permettant d'insérer les nouvelles données dans les tables.
- Les nouvelles données sont saisies au clavier.
- Ces saisies et les mises à jour des tables sont faites dans un bloc PLSQL.
- Vérifier par des requêtes que les nouvelles données sont bien là.

3. BLOC PLSQL - CURSEUR - BOUCLE FOR...
Sélectionner les n premières oeuvres par ordre décroissant des
valeurs.
Remarque : les valeurs existent nécessairement.

code_oeuvre titre valeur

4. BLOC PLSQL - CURSEUR - BOUCLE LOOP ... END LOOP
Artistes ayant le plus grand nombre d'oeuvres répertoriées

nom date_naissance localite pays nb_oeuvres

5. Liste concaténée des artistes par musée, ayant exposé entre deux dates données.

Chapitre 8 : Oracle - Précompilateur C : proc

8.1 - Introduction

Oracle dispose d'un précompilateur C qui permet d'intégrer des commandes SQL dans un programme écrit en C.
Pendant la précompilation, Oracle génère des séquences de code qui insèrent les commandes SQL dans le programme C.
Le nom du fichier source a pour extension .pc.
La précompilation se fait avec la commande

```
proc iname=source.pc [options]
```

Le fichier peut être spécifié sans l'extension. On obtient un fichier de même nom et d'extension .c dans lequel les commandes SQL ont été traduites en instructions du langage C.

8.2 - Structure du programme

Up

```
/*  accès aux fichiers INCLUDE du langage C */
#include
/* partie déclaration */
EXEC SQL BEGIN DECLARE SECTION
- variables hôtes (utilisables dans les commandes SQL avec le préfixe : )
Exemple :
  VARCHAR xchaine(30);
  xchaine est une variable de type structuré :
    struct {unsigned short int len; /* longueur de la chaîne */
                    unsigned char arr[30];
                    }xchaine;
- les variables de connection
    VARCHAR utilisateur(20) = "nomlogin";
    VARCHAR mot_de_passe(20);
EXEC SQL END DECLARE SECTION;
/* inclusion du fichier de communication C - Oracle (bibliothèque proc) */
EXEC SQL INCLUDE SQLCA;
EXEC SQL INCLUDE ORACA;
/* corps du programme C avec la connection à Oracle, les instructions du programme
(instructions C et de commandes SQL (ou blocs PL/SQL)), la validation ou annulation des
actions SQL, la déconnection. */
```

EXEC SQL CONNECT :utilisateur [IDENTIFIED BY :mot_de_passe];

8.3 - Exemple Up

```
/* Define constants for VARCHAR lengths. */
#define     UNAME_LEN      20
#define     PWD_LEN        40

/* acces aux librairies standard d'entree-sorties */
/* equivalent du uses CRT du pascal */

#include

/* declaration des variables oracle/c */
EXEC SQL BEGIN DECLARE SECTION;
char libelle[40];
float valeur;
/* la variable uid sert a se connecter a l'ensemble ORACLE */
VARCHAR uid[UNAME_LEN];
VARCHAR pwd[PWD_LEN];
int counteur;
int compteur;
EXEC SQL INCLUDE ORACA;
EXEC SQL INCLUDE SQLCA;
EXEC SQL END DECLARE SECTION;

/* acces aux librairies Pro*c */
/* On definit en fait dans SQLCA toutes les fonctions et variables qui vont */
/* permettre aux compilateurs proC de traduire ce programme en C tout court */
/* afin qu'un simple cc puisse le compiler */
int compteur;
/* procedure principale et unique */
/* argc est le nombre de parametres passes au programme */
/* argv est le tableau d'emplacement de ces parametres */

void main(argc,argv)
int argc;
char *argv[];

{
```

```
/* connexion a Oracle */
   strncpy((char *) uid.arr, "monet", UNAME_LEN);
   uid.len = strlen((char *) uid.arr);
/* Copy the password.
*    strncpy((char *) pwd.arr, "*****", PWD_LEN);
*   pwd.len = strlen((char *) pwd.arr);
*   EXEC SQL CONNECT :uid IDENTIFIED BY :pwd;
*/
/* Une petite interrogation sur la table A_artiste pour vérifier que l'on
 est bien connecté
*/
EXEC SQL SELECT count(*) into :counteur from A_artiste;
printf("le counteur donne %i\n", counteur);
for (compteur=1; compteur<5;compteur++)
{printf("coucou %i\n", compteur);
}

/* Validation des modifs */
/* EXEC SQL COMMIT;
*/

/* Deconnexion de la base */
/* EXEC SQL DISCONNECT;
*/

EXEC SQL COMMIT WORK RELEASE;

/* Attention aux problemes de Verrou si plusieurs programmes sont lances */
/* en meme temps */
}
```

8.4 - Compilation Up

Version utilisée :
Oracle8 Enterprise Edition Release 8.0.4.0.0 - Production
With the Objects option
PL/SQL Release 8.0.4.0.0 - Production

Pour compiler un fichier de nom nomfic.pc, taper :

make -f demo_proc.mk build OBJS="nomfic.o" EXE=nomfic

Attention ! Ne pas modifier le fichier demo_proc.mk

8.5 - Exercices

Chapître a : Oracle - Quelques commandes utiles - SQLLoader

l. Liste des tables auxquelles on peut accéder :

SELECT owner, table_name

FROM all_tables;

2. Liste des tables personnelles :

SELECT table_name

FROM user_tables;

ou

SELECT table_name

FROM tabs; -- tabs est un synonyme de user_tables.

3. Liste des synonymes de tables :

3.1. Rappelons la structure de la table all_catalog :

SQL>DESC all_catalog (entrée)

Name	Null?	Type
OWNER	NOT NULL	VARCHAR2(30)
TABLE_NAME	NOT NULL	VARCHAR2(30)
TABLE_TYPE		VARCHAR2(11)

===> la colonne TABLE_TYPE permet de connaître le nom des synonymes de tables.

3.2 Sélection des occurrences contenant les noms des synonymes de tables :

SELECT *

FROM all_catalog

WHERE table_type == 'SYNONYM';

==> on peut voir les synonymes des tables 'Artistes'.

Par exemple, A_artiste est un synonyme de la table fm.A_artiste (accessible à tout public en lecture pour les requêtes).

Dans les requêtes, on peut remplacer fm.A_artiste. par son synonyme A_artiste

3.3 Requête sur la table fm.A_artiste à partir de son synonyme :

SELECT *

FROM A_artiste;
==> 52 occurrences

4. Création d'une table A_artiste dans son tablespace personnel :
4.1 Création de la structure :

```
Create Table A_artiste
  (cdart CHAR(2) PRIMARY KEY, /* code artiste*/
  nom VARCHAR (30),   /* Nom artiste */
  sexe CHAR (1),   /* sexe [l, 2] */
  datns VARCHAR ( 10 ) , /* Date ou année de naissance */
  localite VARCHAR(25), /* Localité de naissance */
  pays VARCHAR (15), /* Pays de naissance */
  datdc VARCHAR ( 10 ) , /* Date ou année de décès (sinon jour 1 et mois 1)*/
  cdmtr CHAR ( 2) /* Code maître */
  );
```

====> Cette table a le même nom que le synonyme. Elle masque l'accès à la table fm.A_artiste par son synonyme.

4.2 Contenu de la table A_artiste :

```
  SELECT *
    FROM A_artiste;
```

===> accès à la table de son propre tablespace, donc la table est vide.

```
  SELECT *
    FROM fm.A_artiste;
```

===> on retrouve les 52 occurrences de la table A_artiste existant dans le tablespace de fm.

5. Utilisation du module Sqlloader permettant de récupérer des données en provenance d'une autre Base de Données.

Exercice : Créer et remplir sous Oracle la table A_artiste que l'on avait auparavant sous Access.

5.1 La table Access a été exportée en fichier texte, avec " " comme délimiteurs de champs et des ; comme séparateurs de champs.

5.2 Création de la structure de la table A_artiste (cohérente bien sûr avec les données du fichier texte).

5.3 Adaptation d'un fichier X.ctl à la table que l'on veut récupérer. Voici par exemple le fichier art2.ctl :

```
load data
infile artiste, txt
into table A_artiste
fields terminated by ';' optionally enclosed by ""
(cdart, nom, sexe, datns, localite, pays, datdc, cdmtr)
```

5.4 Lancement **sous unix, en environnement Oracle, mais non sous sqlplus** de la commande :

```
sqlload 'login'/'mot_de_passe' X.ctl
```

==> des fichiers .log et .bad nous renseignent sur les erreurs rencontrées.

Travaux paratique

```
/*
  ===== CAVE D'UN VITICULTEUR ANGEVIN ======
*/

/*
  ===== LES VINS =====
 Les différents vins sont classés.
 - Il existe d'abord des classes de vins (ex. "vin de pays")
 - Chaque classe est divisée en appellations (ex. dans la classe "vin de pays",
 on trouve au moins les appellations "vin de table" et "Vin de Pays de Jardin
 de la France")
 - Chaque appellation est un ensemble de vins nommés (ex. l'appellation dite
 "Appellation d'Origine Controlée" contient les "Bonnezeaux","Anjou Villages",
 "Rosé de Loire", etc. pour ne citer que des vins élaborés chez ce Viticulteur
 Angevin)
 - Chaque vin nommé est subdivisé en "articles" tenant compte de l'année de
 fabrication du vin (le millésime) et du conditionnement (emballage),
*/

/*
  ===== CLIENTS ET TARIFS =====
 La clientèle est divisée en 4 catégories (ou types) : Particuliers, Foyers,
 Societes, Cafe-Hotel-Restaurant.
 Un tarif est adapté à chaque type de client.
*/

/*
  ===== ACHAT =====
 Chaque achat est enregistré sous forme de commande numérotée et datée
*/

/*
 ===== CREATION DES TABLES =====
*/

/*
```

```
 Table des appellations et classes de vins
*/
CREATE TABLE V_type_vins
  (appellation  varchar2(35) PRIMARY KEY, -- une appell. appartient à 1 classe
  classe        varchar2(12)              -- classe de vin = plusieurs appell.
  );

/*
 Table des noms de vins avec leur catégorie (blanc, rosé ou rouge)
*/
CREATE TABLE V_vin
  (nom          varchar2(25) PRIMARY KEY,                                    -- nom du vin (Anjou
Rouge)
  appellation   varchar2(35) REFERENCES V_type_vins (appellation) ,  -- référence à 1 appellat.
  categorie     varchar2(20)                                          -- BLANC, ROUGE ou ROSE
  );

/*
 Table des vins prets à la vente
*/
CREATE TABLE V_article
  (code_article varchar2(8) PRIMARY KEY,                                    -- identifie un vin
millésimé et conditionné
  nom           varchar2(25) REFERENCES V_vin(nom) ON DELETE CASCADE,      -- nom du vin
  millesime     number(2),                                                  -- millésime de ce vin
  emballage     varchar2(10)                                                -- BOUT, LITRES, FILLETTE,
MAGNUM
  );

/*
 Table des types de clientèle
*/
CREATE TABLE V_clientele
  (type_client number(1) PRIMARY KEY,     -- identifiant clientèle : de 1 à 4
  categ_client varchar2(22)               -- Particuliers, Foyers, Sociétés, Hôtels-Restaurants
  );

/*
 Table des références des clients, acheteurs de vin
*/
```

```
CREATE TABLE V_acheteur
 (no_acheteur number PRIMARY KEY,                                  -- identifiant acheteur de vin
 designation  varchar2(40),                                        -- nom, prénom ou label société, ...
 adresse      varchar2(50),                                        -- toute l'adresse dans une chaine
 type_client  number(1) REFERENCES V_clientele(type_client) -- identifie le type du client
 );

/*
 Table des tarifs dépendant des deux identifiants : article-vin et type-client
*/
CREATE TABLE V_tarif
 (code_article varchar2(8) REFERENCES V_article(code_article),          -- identifiant d'un vin
 type_client   number(1)REFERENCES V_clientele(type_client),            -- identifiant clientèle
 prixHT        number(7,2),                                              -- prix dépendant fonct. des
2 ident.
 CONSTRAINT c_tarif PRIMARY KEY (code_article, type_client)
 );

/*
 Table des achats enregistrés
*/
CREATE TABLE V_commande
 (no_commande  number PRIMARY KEY,                    -- identifiant achat d'un client
 date_commande date,                                  -- date achat
 no_acheteur   number REFERENCES V_acheteur(no_acheteur) -- identifiant acheteur
 );

/*
 Table des lignes de commandes pour un article-vin et un no-commande
*/
CREATE TABLE V_article_commande
 (code_article varchar2(8) REFERENCES V_article(code_article), -- identifiant d'un vin
 no_commande   number REFERENCES V_commande(no_commande),      -- identifiant achat
 quantite      number,                                         -- quantité de ce vin ds cet achat
 CONSTRAINT c_V_comm PRIMARY KEY (code_article, no_commande)
 );
```

```
/* insertions des donnees Vins */
/* insertion dans la table V_type_vins */

INSERT INTO V_type_vins VALUES ('Vin de Table','Vin de Pays');
INSERT INTO V_type_vins VALUES ('Vin de Pays de Jardin de la France','Vin de Pays');
INSERT INTO V_type_vins VALUES ('Appellation d''Origine Controlee','A.O.C.');

/* insertion dans la table V_vin */

INSERT INTO V_vin VALUES ('ANJOU BLANC 1/2 SEC','Vin de Table','BLANC');
INSERT INTO V_vin VALUES ('ANJOU BLANC CHENIN','Vin de Pays de Jardin de la
France','BLANC');
INSERT INTO V_vin VALUES ('ANJOU GAMAY','Appellation d''Origine Controlee','BLANC');
INSERT INTO V_vin VALUES ('ANJOU ROUGE','Appellation d''Origine Controlee','ROUGE');
INSERT INTO V_vin VALUES ('ANJOU VILLAGES','Appellation d''Origine Controlee','ROUGE');
INSERT INTO V_vin VALUES ('BONNEZEAUX','Appellation d''Origine Controlee','BLANC');
INSERT INTO V_vin VALUES ('CABERNET D''ANJOU','Appellation d''Origine Controlee','ROSE');
INSERT INTO V_vin VALUES ('COTEAUX DE L''AUBANCE','Appellation d''Origine
Controlee','ROUGE');
INSERT INTO V_vin VALUES ('ANJOU MOUSSEUX','Appellation d''Origine Controlee','BLANC');
INSERT INTO V_vin VALUES ('PINOT BLANC','Vin de Pays de Jardin de la France','BLANC');
INSERT INTO V_vin VALUES ('ROSE D''ANJOU','Appellation d''Origine Controlee','ROSE');
INSERT INTO V_vin VALUES ('ROSE DE LOIRE','Appellation d''Origine Controlee','ROSE');
INSERT INTO V_vin VALUES ('ROUGE VIN DE PAYS','Vin de Table','ROUGE');
INSERT INTO V_vin VALUES ('SAUVIGNON','Vin de Pays de Jardin de la France','BLANC');

/* insertion dans la table V_article */

INSERT INTO V_article VALUES ('ABD93L','ANJOU BLANC 1/2 SEC',93,'LITRES');
INSERT INTO V_article VALUES ('ABC93L','ANJOU BLANC CHENIN',93,'LITRES');
INSERT INTO V_article VALUES ('AG91B','ANJOU GAMAY',91,'BOUT');
INSERT INTO V_article VALUES ('AG93L','ANJOU GAMAY',93,'LITRES');
INSERT INTO V_article VALUES ('AG93B','ANJOU GAMAY',93,'BOUT');
INSERT INTO V_article VALUES ('AR91B','ANJOU ROUGE',91,'BOUT');
INSERT INTO V_article VALUES ('AR91F','ANJOU ROUGE',91,'FILLETTE');
INSERT INTO V_article VALUES ('AR92B','ANJOU ROUGE',92,'BOUT');
INSERT INTO V_article VALUES ('AR92L','ANJOU ROUGE',92,'LITRES');
INSERT INTO V_article VALUES ('AR92F','ANJOU ROUGE',92,'FILLETTE');
INSERT INTO V_article VALUES ('AR93B','ANJOU ROUGE',93,'BOUT');
INSERT INTO V_article VALUES ('AR93L','ANJOU ROUGE',93,'LITRES');
```

```
INSERT INTO V_article VALUES ('AV90','ANJOU VILLAGES',90,'BOUT');
INSERT INTO V_article VALUES ('AV91','ANJOU VILLAGES',91,'BOUT');
INSERT INTO V_article VALUES ('AV92MAG','ANJOU VILLAGES',92,'MAGNUM');
INSERT INTO V_article VALUES ('AV93','ANJOU VILLAGES',93,'BOUT');
INSERT INTO V_article VALUES ('BX89B','BONNEZEAUX',89,'BOUT');
INSERT INTO V_article VALUES ('BX90B','BONNEZEAUX',90,'BOUT');
INSERT INTO V_article VALUES ('BX91B','BONNEZEAUX',91,'BOUT');
INSERT INTO V_article VALUES ('BX92B','BONNEZEAUX',92,'BOUT');
INSERT INTO V_article VALUES ('BX93B','BONNEZEAUX',93,'BOUT');
INSERT INTO V_article VALUES ('BX93L','BONNEZEAUX',93,'LITRES');
INSERT INTO V_article VALUES ('CA92FIL', 'CABERNET D''ANJOU',92,'FILLETTE');
INSERT INTO V_article VALUES ('CA92B', 'CABERNET D''ANJOU',92,'BOUT');
INSERT INTO V_article VALUES ('CA92L', 'CABERNET D''ANJOU',92,'LITRES');
INSERT INTO V_article VALUES ('CA93L', 'CABERNET D''ANJOU',93,'LITRES');
INSERT INTO V_article VALUES ('CXA92B', 'COTEAUX DE L''AUBANCE',92,'BOUT');

INSERT INTO V_article VALUES ('CXA93B', 'COTEAUX DE L''AUBANCE',93,'BOUT');
INSERT INTO V_article VALUES ('CXA93L', 'COTEAUX DE L''AUBANCE',93,'LITRES');
INSERT INTO V_article VALUES ('CPH92B', 'ANJOU MOUSSEUX',92,'BOUT');
INSERT INTO V_article VALUES ('PIN92B', 'PINOT BLANC',92,'BOUT');
INSERT INTO V_article VALUES ('PIN92L', 'PINOT BLANC',92,'LITRES');
INSERT INTO V_article VALUES ('PIN93B', 'PINOT BLANC',93,'BOUT');
INSERT INTO V_article VALUES ('PIN93L', 'PINOT BLANC',93,'LITRES');
INSERT INTO V_article VALUES ('RA92L', 'ROSE D''ANJOU',92,'LITRES');
INSERT INTO V_article VALUES ('RA93L', 'ROSE D''ANJOU',93,'LITRES');
INSERT INTO V_article VALUES ('RL92B', 'ROSE DE LOIRE',92,'BOUT');
INSERT INTO V_article VALUES ('RL92L', 'ROSE DE LOIRE',92,'LITRES');
INSERT INTO V_article VALUES ('RL93L', 'ROSE DE LOIRE',93,'LITRES');
INSERT INTO V_article VALUES ('GR92L', 'ROUGE VIN DE PAYS',92,'LITRES');
INSERT INTO V_article VALUES ('GR93L', 'ROUGE VIN DE PAYS',93,'LITRES');
INSERT INTO V_article VALUES ('GR94L', 'ROUGE VIN DE PAYS',94,'LITRES');
INSERT INTO V_article VALUES ('SAU92B', 'SAUVIGNON',92,'BOUT');
INSERT INTO V_article VALUES ('SAU93L', 'SAUVIGNON',93,'LITRES');
INSERT INTO V_article VALUES ('SAU93B', 'SAUVIGNON',93,'BOUT');

/* table V_clientele = types de clients */

INSERT INTO V_clientele VALUES (1,'Particuliers');
INSERT INTO V_clientele VALUES (2,'Foyers');
INSERT INTO V_clientele VALUES (3,'Societes');
```

```
INSERT INTO V_clientele VALUES (4,'Cafe-Hotel-Restaurant');

/* table V_acheteur = les acheteurs */

INSERT INTO V_acheteur VALUES (1,'marchand catherine','26 rue hoche 45613 courville',1);
INSERT INTO V_acheteur VALUES (2,'fuclaux francois','6 impasse galiieni 63110 pinboche',1);
INSERT INTO V_acheteur VALUES (3,'clovis laurette','31 rue des ursulines 12560 leclos',1);
INSERT INTO V_acheteur VALUES (4,'tordose amelie','33 impasse des fleurs 44620 gorges',1);
INSERT INTO V_acheteur VALUES (5,'tarmiente victor','clos des vignes 69540 beaujolais',1);
INSERT INTO V_acheteur VALUES (6,'vanelant vanessa','42 lotissement des bieres 54770 basse-
yutz',1);
INSERT INTO V_acheteur VALUES (7,'ouagadougou mamadou','57 rue du niger 78950 bievres',1);

INSERT INTO V_acheteur VALUES (8,'la vie en rose','99 impasse geronte 75015 paris',2);
INSERT INTO V_acheteur VALUES (9,'les pins d''alep','62 rue du soudan 21540 gagneux',2);
INSERT INTO V_acheteur VALUES (10,'foyer maurice chevalier','26 avenue chapeau 92126
voigny',2);
INSERT INTO V_acheteur VALUES (11,'association des nonagenaires','13 rue souris 75114 paris',2);
INSERT INTO V_acheteur VALUES (12,'foyer dagobert','1 rue des ceinturons 73610 cotraz',2);
INSERT INTO V_acheteur VALUES (13,'foyer jules marey','23 rue des cellophanes 85220
bayeurley',2);
INSERT INTO V_acheteur VALUES (14,'societe manloxer','51 rue batracien 59450 batardeau',3);
INSERT INTO V_acheteur VALUES (15,'societe duchemole','64 avenue tempion 45780 tartifume',3);
INSERT INTO V_acheteur VALUES (16,'societe eaugrise','26 avenue clairefontaine 88200 saint-
die',3);
INSERT INTO V_acheteur VALUES (17,'societe tonusse','12 rue de padirac 65420 saint-flour',3);
INSERT INTO V_acheteur VALUES (18,'societe vanlaire','89 rue des eoliennes 69550
mangelevent',3);
INSERT INTO V_acheteur VALUES (19,'societe des roses','11 avenue inria 44850 cebonvivre',3);
INSERT INTO V_acheteur VALUES (20,'hotel bellevue','6 rue des fenetres 56254 zieutey',4);
INSERT INTO V_acheteur VALUES (21,'cafe hotel la bonne brise','29440 ouessant',4);
INSERT INTO V_acheteur VALUES (22,'hotel resto au bon coeur','33 rue michel colucci 78230
plaisir',4);
INSERT INTO V_acheteur VALUES (23,'cafe hotel resto a la vieille croute','76550 bonbeurre',4);
INSERT INTO V_acheteur VALUES (24,'pension familia','5 cours du cinematographe 54770
langenfeld',4);
INSERT INTO V_acheteur VALUES (25,'les mimosas','36 avenue du soleil 13200 brabassant',4);

/* table V_tarif (de */
```

```
INSERT INTO V_tarif VALUES ('ABC93L',1,9.69);
INSERT INTO V_tarif VALUES ('ABD93L',1,11.80);
INSERT INTO V_tarif VALUES ('AG91B',1,16.86);
INSERT INTO V_tarif VALUES ('AG93L',1,10.11);
INSERT INTO V_tarif VALUES ('AR91B',1,18.54);
INSERT INTO V_tarif VALUES ('AR92B',1,16.86);
INSERT INTO V_tarif VALUES ('AR92L',1,10.96);
INSERT INTO V_tarif VALUES ('AV90',1,20.23);
INSERT INTO V_tarif VALUES ('AV91',1,21.07);
INSERT INTO V_tarif VALUES ('AV92MAG',1,67.45);
INSERT INTO V_tarif VALUES ('BX89B',1,50.59);
INSERT INTO V_tarif VALUES ('BX90B',1,50.59);
INSERT INTO V_tarif VALUES ('BX91B',1,42.15);
INSERT INTO V_tarif VALUES ('BX92B',1,42.15);
INSERT INTO V_tarif VALUES ('BX93L',1,21.07);
INSERT INTO V_tarif VALUES ('CA92B',1,15.17);
INSERT INTO V_tarif VALUES ('CA93L',1,10.11);
INSERT INTO V_tarif VALUES ('CPH92B',1,26.98);
INSERT INTO V_tarif VALUES ('CXA92B',1,25.29);
INSERT INTO V_tarif VALUES ('CXA93L',1,15.17);
INSERT INTO V_tarif VALUES ('GR93L',1,6.32);
INSERT INTO V_tarif VALUES ('PIN92B',1,16.86);
INSERT INTO V_tarif VALUES ('PIN93L',1,9.69);
INSERT INTO V_tarif VALUES ('RA92L',1,7.58);
INSERT INTO V_tarif VALUES ('RL93L',1,9.27);
INSERT INTO V_tarif VALUES ('SAU92B',1,18.54);
INSERT INTO V_tarif VALUES ('SAU93B',1,18.54);
INSERT INTO V_tarif VALUES ('SAU93L',1,11.80);

INSERT INTO V_tarif VALUES ('AG93B',2,12.94);
INSERT INTO V_tarif VALUES ('AG93L',2,8.43);
INSERT INTO V_tarif VALUES ('AR92B',2,13.78);
INSERT INTO V_tarif VALUES ('AR92F',2,7.58);
INSERT INTO V_tarif VALUES ('AR93B',2,13.78);
INSERT INTO V_tarif VALUES ('AR93L',2,9.27);
INSERT INTO V_tarif VALUES ('AV90',2,16.86);
INSERT INTO V_tarif VALUES ('AV92MAG',2,42.15);
INSERT INTO V_tarif VALUES ('AV93',2,16.86);
INSERT INTO V_tarif VALUES ('BX91B',2,31.19);
INSERT INTO V_tarif VALUES ('BX92B',2,31.19);
```

```
INSERT INTO V_tarif VALUES ('CA92B',2,10.96);
INSERT INTO V_tarif VALUES ('CPH92B',2,26.98);
INSERT INTO V_tarif VALUES ('CXA93B',2,18.54);
INSERT INTO V_tarif VALUES ('CXA93L',2,15.17);
INSERT INTO V_tarif VALUES ('GR94L',2,6.32);
INSERT INTO V_tarif VALUES ('PIN93B',2,10.11);
INSERT INTO V_tarif VALUES ('RA93L',2,7.16);
INSERT INTO V_tarif VALUES ('RL92B',2,10.11);
INSERT INTO V_tarif VALUES ('SAU93B',2,16.86);

INSERT INTO V_tarif VALUES ('ABC93L',3,7.58);
INSERT INTO V_tarif VALUES ('AR91F',3,7.58);
INSERT INTO V_tarif VALUES ('AR92B',3,13.49);
INSERT INTO V_tarif VALUES ('AR92L',3,9.27);
INSERT INTO V_tarif VALUES ('CA92FIL',3,6.74);
INSERT INTO V_tarif VALUES ('CA92B',3,11.8);
INSERT INTO V_tarif VALUES ('CA92L',3,8.43);
INSERT INTO V_tarif VALUES ('CXA92B',3,21.92);
INSERT INTO V_tarif VALUES ('CPH92B',3,26.98);
INSERT INTO V_tarif VALUES ('PIN92L',3,7.58);
INSERT INTO V_tarif VALUES ('RA92L',3,7.16);
INSERT INTO V_tarif VALUES ('GR93L',3,6.32);

INSERT INTO V_tarif VALUES ('AR91B',4,14.33);
INSERT INTO V_tarif VALUES ('AR92B',4,14.33);
INSERT INTO V_tarif VALUES ('AR92L',4,9.27);
INSERT INTO V_tarif VALUES ('AV90',4,16.86);
INSERT INTO V_tarif VALUES ('AV91',4,18.54);
INSERT INTO V_tarif VALUES ('BX91B',4,33.72);
INSERT INTO V_tarif VALUES ('BX92B',4,33.72);
INSERT INTO V_tarif VALUES ('CA92B',4,12.64);
INSERT INTO V_tarif VALUES ('CPH92B',4,26.98);
INSERT INTO V_tarif VALUES ('PIN92B',4,14.33);
INSERT INTO V_tarif VALUES ('RL92B',4,11.8);
INSERT INTO V_tarif VALUES ('RL92L',4,9.27);
INSERT INTO V_tarif VALUES ('GR93L',4,5.90);


/* table V_commande */
```

```
INSERT INTO V_commande VALUES (1,'21-JUN-96',3);
INSERT INTO V_commande VALUES (2,'24-JUN-96',2);
INSERT INTO V_commande VALUES (3,'30-JUN-96',1);
INSERT INTO V_commande VALUES (4,'10-JUL-96',10);
INSERT INTO V_commande VALUES (5,'12-JUL-96',15);
INSERT INTO V_commande VALUES (6,'15-JUL-96',10);
INSERT INTO V_commande VALUES (7,'18-JUL-96',4);
INSERT INTO V_commande VALUES (8,'24-JUL-96',17);
INSERT INTO V_commande VALUES (9,'28-JUL-96',21);
INSERT INTO V_commande VALUES (10,'28-JUL-96',16);
INSERT INTO V_commande VALUES (11,'02-AUG-96',11);
INSERT INTO V_commande VALUES (12,'04-AUG-96',20);
INSERT INTO V_commande VALUES (13,'06-AUG-96',5);
INSERT INTO V_commande VALUES (14,'12-AUG-96',6);
INSERT INTO V_commande VALUES (15,'15-AUG-96',7);
INSERT INTO V_commande VALUES (16,'15-AUG-96',18);
INSERT INTO V_commande VALUES (17,'20-AUG-96',22);
INSERT INTO V_commande VALUES (18,'30-AUG-96',13);
INSERT INTO V_commande VALUES (19,'03-SEP-96',8);
INSERT INTO V_commande VALUES (20,'10-SEP-96',9);
INSERT INTO V_commande VALUES (21,'15-SEP-96',12);
INSERT INTO V_commande VALUES (22,'20-SEP-96',13);
INSERT INTO V_commande VALUES (23,'10-OCT-96',19);
INSERT INTO V_commande VALUES (24,'20-OCT-96',20);
INSERT INTO V_commande VALUES (25,'28-OCT-96',24);
INSERT INTO V_commande VALUES (26,'29-NOV-96',14);
INSERT INTO V_commande VALUES (27,'14-DEC-96',23);
INSERT INTO V_commande VALUES (28,'30-DEC-96',25);
INSERT INTO V_commande VALUES (29,'15-JAN-97',16);
INSERT INTO V_commande VALUES (30,'21-JAN-97',25);
INSERT INTO V_commande VALUES (31,'27-FEB-97',22);


/* table V_article_commande */

INSERT INTO V_article_commande VALUES ('AR92B',1,6);
INSERT INTO V_article_commande VALUES ('SAU92B',1,12);
INSERT INTO V_article_commande VALUES ('AR91B',2,3);
INSERT INTO V_article_commande VALUES ('BX92B',2,3);
INSERT INTO V_article_commande VALUES ('SAU93B',2,3);
```

```
INSERT INTO V_article_commande VALUES ('BX89B',3,6);
INSERT INTO V_article_commande VALUES ('CA93L',3,32);
INSERT INTO V_article_commande VALUES ('AR92F',4,12);
INSERT INTO V_article_commande VALUES ('CXA93L',4,32);
INSERT INTO V_article_commande VALUES ('SAU93B',4,6);
INSERT INTO V_article_commande VALUES ('ABC93L',5,32);
INSERT INTO V_article_commande VALUES ('CA92B',5,6);
INSERT INTO V_article_commande VALUES ('PIN92L',5,10);
INSERT INTO V_article_commande VALUES ('CPH92B',6,10);
INSERT INTO V_article_commande VALUES ('BX89B',7,6);
INSERT INTO V_article_commande VALUES ('BX90B',7,6);
INSERT INTO V_article_commande VALUES ('AR91F',8,12);
INSERT INTO V_article_commande VALUES ('AR91B',9,12);
INSERT INTO V_article_commande VALUES ('AV90',9,24);
INSERT INTO V_article_commande VALUES ('BX92B',9,12);
INSERT INTO V_article_commande VALUES ('GR93L',9,64);
INSERT INTO V_article_commande VALUES ('AR92B',10,6);
INSERT INTO V_article_commande VALUES ('PIN92L',10,32);

INSERT INTO V_article_commande VALUES ('AR93B',11,12);
INSERT INTO V_article_commande VALUES ('AV90',11,12);
INSERT INTO V_article_commande VALUES ('GR94L',11,64);
INSERT INTO V_article_commande VALUES ('CA92B',12,6);
INSERT INTO V_article_commande VALUES ('BX92B',12,24);
INSERT INTO V_article_commande VALUES ('CPH92B',12,12);
INSERT INTO V_article_commande VALUES ('RL92L',12,64);
INSERT INTO V_article_commande VALUES ('AR91B',13,12);
INSERT INTO V_article_commande VALUES ('PIN92B',14,6);
INSERT INTO V_article_commande VALUES ('SAU92B',14,6);

INSERT INTO V_article_commande VALUES ('BX93L',15,10);
INSERT INTO V_article_commande VALUES ('CA92L',16,32);
INSERT INTO V_article_commande VALUES ('AR92L',16,32);
INSERT INTO V_article_commande VALUES ('AR92L',17,64);
INSERT INTO V_article_commande VALUES ('GR93L',17,128);
INSERT INTO V_article_commande VALUES ('AV93',18,6);
INSERT INTO V_article_commande VALUES ('CA92B',18,12);
INSERT INTO V_article_commande VALUES ('AR93L',19,32);
INSERT INTO V_article_commande VALUES ('CXA93L',19,32);
INSERT INTO V_article_commande VALUES ('AV92MAG',20,16);
```

```
INSERT INTO V_article_commande VALUES ('GR94L',21,32);
INSERT INTO V_article_commande VALUES ('RA93L',21,32);
INSERT INTO V_article_commande VALUES ('SAU93B',21,6);
INSERT INTO V_article_commande VALUES ('AR93B',22,6);
INSERT INTO V_article_commande VALUES ('GR94L',22,32);
INSERT INTO V_article_commande VALUES ('AR92B',23,12);
INSERT INTO V_article_commande VALUES ('CA92B',23,12);
INSERT INTO V_article_commande VALUES ('PIN92L',23,10);
INSERT INTO V_article_commande VALUES ('BX91B',24,24);
INSERT INTO V_article_commande VALUES ('BX92B',24,24);
INSERT INTO V_article_commande VALUES ('PIN92B',24,12);
INSERT INTO V_article_commande VALUES ('AR92L',25,32);
INSERT INTO V_article_commande VALUES ('RL92L',25,32);
INSERT INTO V_article_commande VALUES ('CA92L',26,48);
INSERT INTO V_article_commande VALUES ('AR92L',27,64);
INSERT INTO V_article_commande VALUES ('GR93L',27,32);

INSERT INTO V_article_commande VALUES ('BX91B',28,12);
INSERT INTO V_article_commande VALUES ('BX92B',28,12);
INSERT INTO V_article_commande VALUES ('PIN92B',28,6);
INSERT INTO V_article_commande VALUES ('PIN92L',29,32);
INSERT INTO V_article_commande VALUES ('RA92L',29,32);
INSERT INTO V_article_commande VALUES ('CA92B',30,6);
INSERT INTO V_article_commande VALUES ('AV91',30,12);
INSERT INTO V_article_commande VALUES ('AR92L',31,32);
INSERT INTO V_article_commande VALUES ('RL92L',31,32);
```

```
/* suppression des tables vins : l'ordre de suppression dépend des dépendances fonctionnelles */

DROP TABLE V_article_commande;

DROP TABLE V_commande;

DROP TABLE  V_tarif;

DROP TABLE V_acheteur;

DROP TABLE V_clientele;

DROP TABLE V_article;

DROP TABLE V_vin;

DROP TABLE V_type_vins;
```

/ * TP2 - Vins - Requêtes de base, jointures, ordre */

SUJET

Une cave viticole en Anjou

REQUETES

1. Afficher les noms des vins blancs (de categorie BLANC).

NOM

-- rangement des résultats au niveau de l'affichage

2. Afficher la liste des appellations et des noms des vins ordonnés selon l'appellation.

APPELLATION NOM

-- Jointure et rangement des résultats

3. Afficher la liste des noms de vins, du conditionnement, de la catégorie ordonnés selon la catégorie et dans chaque catégorie selon le conditionnement.

NOM EMBALLAGE CATEGORIE

4. Afficher la liste des clients avec le numéro et la date de leurs commandes. Ordonner les résultats selon la désignation du client et la date de commande.

DESIGNATION NO_COMMANDE DATE_COMMANDE

-- colonne calculée, alias de colonne, (arrondi ; voir chapître sur les fonctions

5. Afficher le nom du vin, le millésime, le prix TTC pour les vins en bouteille vendus aux particuliers.
Ordonner les résultats selon le nom, le millésime (ordre décroissant).

NOM MILLESIME PRIX

-- Utilisation des opérateurs de l'algèbre relationnelle

6. Déterminer les vins (nom, millésime, conditionnement) vendus aux Particuliers et non aux Sociétés.
Ordonner selon le conditionnement, le nom.

EMBALLAGE NOM MILLESIME

/ * TP3 - Vins - Fonctions, Agrégation de données, Mise en page des résultats */

SUJET

Une cave viticole en Anjou

REQUETES

1. Afficher le montant de la vente TTC de chaque vin (defini par son nom).

2. Afficher le montant de la vente TTC par catégorie de vin.

3. Afficher la liste des désignations des acheteurs 'Particuliers' avec le montant global de leurs commandes.

4. Afficher le nombre de litres de vin vendus, pour chaque vin (défini par son nom, son millésime), sachant qu'une bouteille contient 0,75 litre, une fillette 0,375 litre et un magnum 1,5 litre.
Le nom du vin ne sera pas répété.

5. Liste des différents noms de vins classés par catégorie et rangés dans l'ordre alphabétique. A l'affichage, la catégorie n'est pas répétée, une ligne blanche sépare les catégories.

6. Calculer le montant TTC des ventes par année civile.
Affichage par ordre chronologique des années.

-- Utiliser Having
7. Afficher les prix moyens HT des différentes vins définis par leur nom et conditionnés en bouteilles sachant que l'on s'intéresse à une seule catégorie d'acheteurs.

/ * TP4 - Vins -Sous-Requêtes /

REQUETES

1. Existe-t-il des vins (nom, millésime, conditionnement) vendus exclusivement
à une catégorie de clients ? Si oui, en établir la liste par type de clientèle.

2. Calculer le montant TTC des ventes, par année civile.

3. Afficher le nom, le millésime, le nombre de bouteilles de l'article
(conditionné en bouteille) le plus commandé.
Remarque : en réponse, il peut y avoir plusieurs articles.

--Vues
4. Nous voulons faire parvenir une publicite aux cafes-hotels-restaurant en leur
envoyant le tarif HT qui leur convient avec le code_article, le nom, le millesime,
l'emballage, le prix HT.Creer une vue contenant tous ces renseignements.

5. Nous voulons établir la facture correspondant a un numéro de commande donné.
Créer une vue pour chaque point suivant :
- les coordonnées de l'acheteur
- l'ensemble des lignes de la facture avec : nom du vin, millesime, quantite, emballage,
montant TTC.
Déterminer le montant total de la facture.

6. Afficher, pour les acheteurs particuliers et pour le dernier millésime pour lequel
des particuliers ont fait des achats, la liste des noms de vin, leur conditionnement
(emballage), leur prix TTC (la TVA est a 19.6%).
Ranger les résultats selon la categorie et le nom du vin.
Faire une rupture avec saut de ligne pour chaque catégorie.

-- dates, sous-requête, fonctions
7. Liste des clients avec la date de leur dernière commande.

-- date, saisie au clavier, sous-requête
8. On veut envoyer un encart publicitaire aux clients qui n'ont rien commandé depuis
une date donnée.
Afficher la liste de ces clients (avec leurs coordonnées).

-- fonctions, vues

8. On veut faire profiter d'une promotion les clients qui ont passé commande pour plus de X Francs depuis une date donnée.
La date et la valeur X sont saisies au clavier.
Afficher la liste de ces clients (avec leurs coordonnées).