SGBD TP5 :

PL/SQL : SELECT INTO, CURSEUR (implicite et explicite).

- Durée 1h30.
- Pour ce TP vous avez besoin de la BD Gest_Appart (G_Appartement.txt) et du fichier de données (donneesGApp.txt)
- Le compte redu de ce TP doit être sous la forme suivante : n°question, bloc PL/SQL, capture écran du résultat.

Le schéma relationnel de la BD Gest_Appart :

Immeuble (idimm, adresse, codep, nbrEtages, dateConstr)

Personne (idpers, nom, prenom, adrOfficielle, codep, dateN, profession, tel, Persmorale)

Appartement (numapp, #idimm, descript, nbrpiece, superficie, etage, occupe)

ContratAchat(numcontA, #(numapp, idimm), dateAchat, PrixAchat)

PropAppart (#numcontA, #idpers)

ContratLoc (numcontLoc, #(numapp, idimm), dateLoc, #idpers, datedepart, datedepR, loyer,Caution)

Remarques :
PropAppart : *Un appartement peut avoir plusieurs propriétaire**s.***
Appartement.occupe : *Cet attribut prend la valeur 'oui' s'il est actuellement occupé et la valeur 'non' une fois qu'il ne l'est plus.*
Personne.Persomorale : *Cet attribut prend la valeur 'non' si la personne n'est pas une personne physique (ex : société, association..).*
ContratLoc.datedepart : *pour une location d'un appartement on doit préciser la date de début de la location et la date de départ prévue.*
ContratLoc.datedepR : *Cet attribut prend comme valeur la date du départ réel d'une personne. La datedepR peut être inferieur à la datedepart si la personne quitte l'appartement avant la période prévue de la location.*

Travail à faire :

1. Créer la base de données **Gest_Appart** (voir le script du fichier G_Appartement.txt) et Insérer les données dans les tables (voir le script du fichier donneesGApp.txt).

2. Ecrire un bloc PL/SQL qui va chercher le nombre d'appartements libres ayants 3 pièces.

```
Résultat d'exécution :
                          *** Appartements disponibles avec: 3 pieces***

                          Il y'a 2 appartements
```

3. Le bloc PL/SQL ci-dessous permet de chercher le propriétaire de l'appartement numéro 1 de l'immeuble 12.

```
DECLARE
Vnom personne.nom%TYPE;
Vprenom personne.prenom%TYPE ;
Vidprop personne.idpers%TYPE ;

VNumapp contratachat.numapp%TYPE :=1 ;
vIdimm contratachat.idimm%TYPE :=12 ;
vnbrcont NUMERIC;

BEGIN
--verifier si cet appartement se trouve dans la table contratAchat

SELECT COUNT(numcontA) INTO vnbrcont
FROM contratachat
WHERE numapp=vnumapp AND idimm=vidimm ;
```

```
--traitement à faire si cet appartement existe dans la table contratAchat
IF vnbrcont>0 THEN

SELECT p.idpers, nom, prenom INTO vidprop,vnom, vprenom
FROM personne p, contratachat ca, propappart pa
WHERE numapp=vnumapp AND idimm=vidimm
AND p.idpers=pa.idpers
AND ca.numconta=pa.numconta

-- chercher le dernier contrat d''achat concernant cet appartement

AND dateAchat = (SELECT MAX(dateachat) FROM contratachat
WHERE numapp=vnumapp AND idimm=vidimm);

      DBMS_OUTPUT.PUT_LINE ('le proprietaire de l''appartement est :  '
      || vidprop ||' '|| vnom ||' '||vprenom ) ;

      DBMS_OUTPUT.NEW_LINE ; -- Sauter une ligne

ELSE
DBMS_OUTPUT.PUT_LINE ('Aucun appartement trouvé avec ces critères!');
END IF;
END ;
```

3.1. Avec le même bloc PL/SQL chercher le propriétaire de l'appartement numéro 1 de l'immeuble 13. Expliquer le résultat de l'exécution.

3.2. On veut écrire un bloc PL/SQL qui permet de chercher le ou les propriétaires d'un appartement vu qu'un appartement peut avoir plusieurs propriétaires à une date donnée.

*ⓘ Avant la première extraction, **nomCurseur%NOTFOUND** revoie toujours **NULL.** Si l'instruction **FETCH** ne parvient jamais à s'exécuter correctement, la boucle répéter devient infinie.*

*=> Il est conseillé de programmer la sortie d'une structure répéter à l'aide de la condition composée : **EXIT WHEN nomCurseur%NOTFOUND OR nomCurseur%NOTFOUND IS NULL.***

Résultat d'exécution :

```
Liste des proprietaires de l'appartement:
-9 BENARFA Ahmed
-10 BENAhmed Anissa
```

4. On veut ecrire un bloc PL/SQL qui augmente de 5% les loyers des appartements, loués avant le 01/01/18 et qui sont toujours occupés à la date d'aujourd'hui (datedepR IS NULL) :

Résultat d'exécution :

```
1 loyers mis à jour !
```

4.1. Utiliser un curseur Implicite.

4.2. Utiliser un curseur explicite avec la clause FOR UPDATE.

ⓘ - Une validation (COMMIT) avant la fermeture d'un curseur FOR UPDATE déclenchera une erreur.

- Il n'est pas possible de déclarer un curseur FOR UPDATE en utilisant dans la requête les directives DISTINCT ou GROUP BY, un opérateur ensembliste, ou une fonction d'agrégat.

5. Le bloc ci dessous permet de chercher le nombre d'appartements non occupés dans chaque immeuble.

```
DECLARE
CURSOR curapplib IS
SELECT im.idimm, adresse, COUNT(ap.idimm)
FROM immeuble im, appartement ap
WHERE im.idimm=ap.idimm
AND UPPER(occupe)=UPPER('non')
GROUP BY im.idimm, adresse;

--on doit definir chaque variable pour recuperer les donnees du cursseur car on a le
--resultat d''un agregat

vidimm immeuble.idimm%TYPE ;
vadresse immeuble.adresse%TYPE ;
vnbrapp NUMERIC ;

BEGIN
OPEN curapplib;

LOOP
FETCH curapplib INTO vidimm,vadresse,vnbrapp;

EXIT WHEN curapplib%NOTFOUND OR curapplib%NOTFOUND IS NULL ;

 DBMS_OUTPUT.PUT_LINE ('-'||vidimm||', '||vadresse||', ' ||vnbrapp||' appartements libres');
 DBMS_OUTPUT.NEW_LINE ;

END LOOP ;

IF curapplib%ROWCOUNT =0 THEN
  DBMS_OUTPUT.PUT_LINE ('Tout les appartements sont occupés ! ') ;
END IF ;
CLOSE curapplib;

END ;
```

5.1. Chercher le nombre et la liste des appartements non occupés pour un immeuble donné (utiliser un curseur paramétré et une boucle WHILE).

Résultat d'exécution :

```
Liste des appartements disponibles au : 5 rue des jasmins 3467

date de construction : 12/04/1941, Nombre d'etages: 5

-1 retapé à neuf 3 150 5

-2 retapé à neuf 3 90 2
```