TD-TP DE BASES DE DONNEES
SEANCE 5 : PROGRAMMATION SQL – PL/SQL – OCILIB

Q1 : Phase préliminaire et prise en main

Créer une table RESULTAT (CODE number, MESSAGE char(50)).
Compléter le programme PL/SQL ci-dessous pour qu'il génère le résultat suivant.

PROGRAMME PL/SQL

```
DELETE FROM RESULTAT ;
PROMPT Nombre de lignes a produire
ACCEPT n
DECLARE
        x NUMBER:= ????????;
BEGIN
        FOR i IN 1..&n LOOP
          IF MOD(x, ????????) = 0 THEN          -- x is even
           INSERT INTO RESULTAT VALUES (?????????????);
          ELSE
           INSERT INTO RESULTAT VALUES (?????????????);
          END IF;
          x := x ????????????? ;
        END LOOP;
        COMMIT;
END;
/
SELECT * FROM RESULTAT ;
```

Contenu table RESULTAT après exécution

CODE	MESSAGE
1	100 is even
2	101 is odd
3	102 is even
… etc	

```
CONNECT system/oracle;
DROP TABLE RESULTAT;
CREATE TABLE RESULTAT (CODE number, MESSAGE char(50));

DELETE FROM RESULTAT ;
PROMPT Nombre de lignes a produire
ACCEPT n
DECLARE
        x NUMBER:= 100;
BEGIN
        FOR i IN 1..&n LOOP
          IF MOD(x, 2) = 0 THEN          -- x is even
           INSERT INTO RESULTAT VALUES (i, x || ' is even');
          ELSE
           INSERT INTO RESULTAT VALUES (i, x || ' is odd');
          END IF;
          x := x + 1;
        END LOOP;
COMMIT;
END;
/
SELECT * FROM RESULTAT ;
```

Q2 : Bloc PL/SQL, expression conditionnelle

Charger le scripts CREATE-bis.sql et CREATE-bis-data.sql. Ecrire un programme PL/SQL qui :
a) demande un numéro de client
b) insère un tuple dans la table RESULTAT et le visualise. Ce tuple est tel que :

- s'il n'y a pas de commandes pour ce client, il a comme valeur 'pas de commande'
- sinon, il a comme valeur 'n commandes pour le client X', avec n le nombre de commandes pour X.

```
PROMPT Numero de client :
ACCEPT n
DECLARE
        nb NUMBER:=0;
BEGIN
        SELECT count(*) into nb from COM WHERE NumCli = &n;
        IF nb = 0 THEN
         INSERT INTO RESULTAT VALUES (1, 'pas de commande');
```

```
        ELSE
         INSERT INTO RESULTAT VALUES (1, nb || ' commandes pour le vendeur &n ');
        END IF;
END;
/
SELECT * FROM RESULTAT;
```

Q3 : Bloc PL/SQL, curseurs et boucles

Ecrire un programme qui permette d'afficher les nième et n+1ième commandes les plus récentes (afficher le numéro de la commande, la date, et le prix total) et d'insérer les informations correspondantes dans la table RESULTAT. Le prix total de la commande sera calculé à partir des détails et des prix des produits commandés. Le nombre *n* est un paramètre saisi par l'utilisateur.

```
PROMPT entre le n pour la nieme commande:
ACCEPT n
DECLARE
    i NUMBER := 0;
BEGIN
    FOR tuple in (      SELECT COM.NumCom, AnCom, sum(prixunit*qte*(1-remise/100)) tot
                        FROM COM, DET, PRO
                        WHERE COM.NumCom = DET.NumCom AND DET.NumPro=PRO.NumPro
                        GROUP BY COM.NumCom, AnCom
                        ORDER BY AnCom, COM.NumCom DESC ) LOOP
            i := i+1;
            IF i = &n OR i = &n+1 THEN
                        INSERT INTO RESULTAT
                        VALUES(i, 'numcom=' || tuple.NumCom || ' total='|| tuple.tot || ' date='|| tuple.ancom);
            END IF;
    END LOOP;
END;
/
SELECT * FROM RESULTAT;
```

Q4 : Programmation OCI vs. PL/SQL vs. SQL

Programmer avec en C avec OCILIB la jointure des tables CLI, COM, DET de la manière suivante:

```
Stocker les clients dans une variable CURSOR nommée CLI_RES ;
Stocker les commandes dans une variable CURSOR nommée COM_RES ;
Stocker les détails  dans une variable CURSOR nommée DET_RES ;
Pour chaque ligne CLI_RES_i de CLI_RES FAIRE :
  NB = 0;
  Pour chaque ligne COM_RES_j de COM_RES FAIRE :
    Pour chaque ligne DET_RES_k de DET_RES FAIRE :
      Si (  CLI_RES_i.Numcli=COM_RES_j.Numcli et
            COM_RES_j.Numcom=DET_RES_k.numcom )
        NB = NB + DET_RES_k.Qte;
      Fin Si;
    Fin Pour;
  Fin Pour;
RES = "nombre des articles du client" + CLI_RES_i.ID + "=" + NB ;
Insérer (CLI_RES_i.ID, RES) dans la table RESUTLAT ;
Fin Pour;
```

Mesure le temps d'exécution du programme OCILIB.

Refaire le programme en PLSQL puis implanter la jointure en SQL pur et mesurer à nouveau les temps d'exécution.

Q5 : Bloc PL/SQL, curseurs paramétrés

Ecrire un programme qui utilisera un curseur paramétré et qui permette de vérifier que le montant payé (COM.Payement) pour une commande donnée (renseignée par l'utilisateur) est égal à la somme des lignes de commandes correspondantes. Pour la commande demandée, une ligne sera insérée dans la table résultat. Cette ligne aura la forme suivante : « NumCom : 100 - Payement : 5000 - Prix : 5000 ».

```
PROMPT Numero de Commande ?
ACCEPT n
```

```
DECLARE
  CURSOR C1 (nc IN NUMBER) IS
    SELECT C.NumCom, C.DateCom, Payement, Sum(P.PrixUnit*D.Qte-D.Remise) PrixTotal
    FROM Com C, Det D, Pro P
    WHERE C.NumCom = D.NumCom AND
          D.NumPro = P.NumPro AND
          C.NumCom = nc
    GROUP BY C.NumCom, C.DateCom, C.Payement;
  CodeUnique resultat.Code%TYPE ;
BEGIN
  SELECT MAX(Code)+1 INTO CodeUnique FROM resultat;
  FOR enr_com IN C1(&n) LOOP
     IF enr_com.Payement != enr_com.PrixTotal THEN
        INSERT INTO resultat VALUES (CodeUnique, 'NumCom:'||enr_com.NumCom||
                          '-Payement:'||enr_com.Payement||'-Prix:'||enr_com.PrixTotal);
        CodeUnique := CodeUnique+1;
     END IF;
  END LOOP;
END;
/
```

Q6 : Bloc PL/SQL, exceptions

Compléter le programme de la question 3 pour gérer l'erreur survenant dans le cas où le nombre N serait strictement supérieur au nombre de commandes dans la table Commandes. Dans ce cas, un message d'erreur sera inséré dans la table résultat.

```
DELETE FROM RESULTAT;
PROMPT entre le n pour la nieme commande:
ACCEPT n
DECLARE
      i NUMBER := 0;
      out_of_range EXCEPTION;
BEGIN
      select count(*) into i from COM;
      IF i <= &n THEN RAISE out_of_range;
      ELSE i := 0; END IF;
      FOR tuple in (          SELECT COM.NumCom, AnCom, sum(prixunit*qte*(1-remise/100)) tot
                              FROM COM, DET, PRO
                              WHERE COM.NumCom = DET.NumCom AND DET.NumPro=PRO.NumPro
                              GROUP BY COM.NumCom, AnCom
                              ORDER BY AnCom, COM.NumCom DESC ) LOOP
                  i := i+1;
                  IF i = &n OR i = &n+1 THEN
                              INSERT INTO RESULTAT
                              VALUES(i, 'numcom=' || tuple.NumCom || ' total='|| tuple.tot || ' date='|| tuple.ancom);
                  END IF;
      END LOOP;
      EXCEPTION
                  WHEN out_of_range THEN
                    BEGIN
                              INSERT INTO resultat VALUES (666, 'Moins de '||&n||' commandes...');
                     END;
END;
/
SELECT * FROM RESULTAT;
```

Q7 : Bloc PL/SQL, packages

Ecrire le package « client » comportant une procédure insérant dans la table RESULTAT les nième et n+1ième commandes d'un client donné en argument et gérant l'exception, et une fonction renvoyant le nombre de commandes pour un client donné passé en argument. Tester l'utilisation du package et des procédures.

```
CREATE OR REPLACE PACKAGE client IS
      FUNCTION nb_com_cli (ncli IN NUMBER);
      PROCEDURE com_n (n IN NUMBER);
      out_of_range EXCEPTION;
END vendor;
/
CREATE OR REPLACE PACKAGE BODY vendor IS
FUNCTION nb_com_cli (ncli IN NUMBER) RETURN NUMBER IS
```

```
	nb NUMBER;
	BEGIN
		SELECT count(*) into nb from COM WHERE NumCli = ncli;
		RETURN nb;
	END;
PROCEDURE com_n (n IN NUMBER) IS
	i NUMBER := 0;
	out_of_range EXCEPTION;
	BEGIN
		select count(*) into i from COM;
		IF i <= n THEN RAISE out_of_range;
		ELSE i := 0;
		END IF;
		FOR tuple in (	SELECT COM.NumCom, AnCom, sum(prixunit*qte*(1-remise/100)) tot
				FROM COM, DET, PRO
				WHERE COM.NumCom = DET.NumCom AND DET.NumPro=PRO.NumPro
				GROUP BY COM.NumCom, AnCom
				ORDER BY AnCom, COM.NumCom DESC ) LOOP
			i := i+1;
			IF i = n OR i = n+1 THEN
				INSERT INTO RESULTAT
				VALUES(i, 'numcom=' || tuple.NumCom || ' total='|| tuple.tot || ' date='|| tuple.ancom);
			END IF;
		END LOOP;
		EXCEPTION
			WHEN out_of_range THEN
				BEGIN
				INSERT INTO resultat VALUES (666, 'Moins de '||n||' commandes...');
				END;
	END;
END vendor;
/

DELETE FROM RESULTAT;
SELECT client.nb_com_cli(10) FROM DUAL;
EXECUTE client.com_n(1000);
EXECUTE client.com_n(5000);
SELECT * FROM RESULTAT;
```