Exercices

Ces exercices permettent de mettre en pratique les principes du langage PL/SQL décrits au cours des séances. Il est recommandé d'enregistrer les bloc PL/SQL dans des fichiers nommé avec l'extension `.sql`.

Exercice 1

Les principes, qui sont tous nécessaires pour créer un bloc PL/SQL simple, incluent la définition des types de données et des identificateurs, ainsi que la validation des expressions.
Dans cet exercice vous allez :

- *déterminer la validité de quelques déclarations*
- *déclarer un bloc PL/SQL simple*
- *exécuter un bloc PL/SQL simple*

1. Evaluez chacune des déclarations suivantes. Déterminez celles qui *ne* sont *pas* valides et expliquez pourquoi.

a.
```
DECLARE
 v_id NUMBER(4);
```

b.
```
DECLARE
 v_x, v_y, v_z VARCHAR2(10);
```

c.
```
DECLARE
 v_birthdate DATE NOT NULL;
```

d.
```
DECLARE
 v_in_stock BOOLEAN := 1;
```

2. Dans chacune des affectations suivantes, indiquez si l'instruction est valide et quel est le type de données du résultat.

a. `v_days_to_go := v_due_date - SYSDATE;`

b. `v_sender := USER || ': ' || TO_CHAR(v_dept_no);`

c. `v_sum := $100,000 + $250,000;`

d. `v_flag := TRUE;`

e. `v_n1 := v_n2 > (2 * v_n3);`

f. `v_value := NULL;`

3. Créez un bloc anonyme qui affiche à l'écran l'expression "Mon bloc PL/SQL fonctionne".

4. Créez un bloc qui déclare deux variables. Affectez la valeur de ces variables à des variables hôte, puis affichez à l'écran les résultats des variables PL/SQL. Exécutez le bloc PL/SQL.

```
V_CHAR Character (variable length)
V_NUM Number
```

Affectez les valeurs suivantes à ces variables :

```
Variable Value
-------- -----------------------------------
V_CHAR The literal '42 is the answer'
V_NUM  The first two characters from V_CHAR
```

Exercice 2

Cet exercice s'appuie sur des exemples de blocs PL/SQL et permet de tester la compréhension des règles de portée. Vous allez :

- *revoir les règles de portée et d'imbrication*
- *développer et tester des blocs PL/SQL*

Bloc PL/SQL

```
DECLARE
v_weight      NUMBER(3) := 600;
v_message   VARCHAR2(255) := 'Product 10012';
BEGIN

 DECLARE
 v_weight    NUMBER(3) := 1;
 v_message   VARCHAR2(255) := 'Product 11001';
 v_new_locn  VARCHAR2(50) := 'Europe';
 BEGIN
 v_weight := v_weight + 1;
 v_new_locn := 'Western ' || v_new_locn;

 END;
v_weight := v_weight + 1;
v_message := v_message || ' is in stock';
v_new_locn := 'Western ' || v_new_locn;

END;
/
```

1. Examinez le bloc PL/SQL ci-dessus et déterminez le type de données et la valeur de chacune des variables suivantes d'après les règles de portée :

a.La valeur de `V_WEIGHT` à la position 1 est :
b.La valeur de `V_NEW_LOCN` à la position 1 est :
c.La valeur de `V_WEIGHT` à la position 2 est :
d.La valeur de `V_MESSAGE` à la position 2 est
e. La valeur de `V_NEW_LOCN` à la position 2 est :

Exemple sur la portée

```
 DECLARE
  v_customer VARCHAR2(50) := 'Womansport';
  v_credit_rating VARCHAR2(50) := 'EXCELLENT';
 BEGIN
DECLARE
 v_customer NUMBER(7) := 201;
 v_name VARCHAR2(25) := 'Unisports';
BEGIN
```

```
 v_customer       v_name v_credit_rating
END;
 v_customer          v_name v_credit_rating
END;
/
```

2. Supposons que vous imbriquiez un sous-bloc dans un bloc, comme indiqué ci-dessus. Vous déclarez deux variables, V_CUSTOMER et V_CREDIT_RATING, dans le bloc principal. Vous déclarez aussi deux variables, V_CUSTOMER et V_NAME, dans le sous-bloc. Déterminez les valeurs et les types de données dans chacun des cas suivants :

a. La valeur de V_CUSTOMER dans le sous-bloc est :
b. La valeur de V_NAME dans le sous-bloc est :
c. La valeur de V_CREDIT_RATING dans le sous-bloc est :
d. La valeur de V_CUSTOMER dans le bloc principal est :
e. La valeur de V_NAME dans le bloc principal est :
f. La valeur de V_CREDIT_RATING dans le bloc principal est :

3.Créez et exécutez un bloc PL/SQL qui accepte deux valeurs numériques via des variables de substitution.
a. Utilisez la commande DEFINE pour indiquer les deux valeurs.

```
DEFINE p_num1 = 2
DEFINE p_num2 = 4
```

b. Transmettez au bloc PL/SQL, via des variables de substitution, les deux valeurs définies au cours de l'étape a). Il faut diviser la première valeur numérique par la seconde, et ajouter cette dernière au résultat. Celui-ci doit être stocké dans une variable PL/SQL et s'afficher à l'écran.
Remarque : Placez la commande SET VERIFY OFF avant le bloc PL/SQL.

4.Créez un bloc PL/SQL qui calcule la rémunération totale pour une année.
a. Les valeurs correspondant au salaire annuel et au pourcentage de prime annuelle sont définies à l'aide de la commande DEFINE.
b. Transmettez au bloc PL/SQL, via des variables de substitution, les valeurs définies au cours de l'étape précédente. La prime doit être convertie d'un nombre entier en nombre décimal (par exemple, de 15 en 0,15). Si le salaire a la valeur null, affectez-lui la valeur zéro avant de calculer la rémunération totale. Exécutez le bloc PL/SQL. *Rappel :* Utilisez la fonction NVL pour traiter les valeurs null.
Remarque : La rémunération totale est égale à la somme du salaire annuel et de la prime annuelle.
Pour tester la fonction NVL, affectez la valeur NULL à la variable de la commande DEFINE.

```
DEFINE p_salary = 50000
DEFINE p_bonus = 10
```

Exercice 3

Dans cet exercice, vous allez écrire des blocs PL/SQL pour sélectionner, entrer, mettre à jour et supprimer les informations contenues dans une table, en utilisant les instructions SQL et les instructions LMD de base dans un bloc PL/SQL.

1. Créez un bloc PL/SQL qui sélectionne dans la table DEPARTMENTS le plus grand numéro de service, et stocke ce dernier dans une variable hote. Affichez le résultat à l'écran.

2. Modifiez le bloc PL/SQL créé au cours de l'exercice 1 pour insérer un nouveau service dans la table DEPARTMENTS.

a. Utilisez la commande DEFINE pour indiquer le nom du service. Nommez le nouveau service Education.
b. Transmettez cette valeur au bloc PL/SQL à l'aide d'une variable de substitution. Plutôt que d'afficher le numéro de service extrait dans l'exercice 1, ajoutez 10 à cette valeur et utilisez-la en tant que numéro du nouveau service.
c. Gardez pour l'instant la valeur NULL en tant que numéro d'emplacement.
d. Exécutez le bloc PL/SQL.
e. Affichez le nouveau service que vous avez créé.

3. Créez un bloc PL/SQL qui met à jour l'ID d'emplacement du nouveau service ajouté au cours de l'exercice précédent.
a. Utilisez une variable hote pour représenter le numéro de service ajouté au cours de l'exercice précédent..
b. Utilisez la commande DEFINE pour indiquer l'ID d'emplacement. Affectez à ce dernier la valeur 1700.

```
DEFINE p_deptno = 280
DEFINE p_loc = 1700
```

c. Transmettez cette valeur au bloc PL/SQL à l'aide d'une variable de substitution. Testez le bloc PL/SQL.
d. Affichez le service mis à jour.

4. Créez un bloc PL/SQL qui supprime le service créé au cours de l'exercice 2.
a.Utilisez la commande DEFINE pour indiquer le numéro de service.

```
DEFINE p_deptno=280
```

b. Transmettez cette valeur au bloc PL/SQL à l'aide d'une variable de substitution. Affichez à l'écran le nombre de lignes affectées.
c. Testez le bloc PL/SQL.
d. Vérifiez que le service a bien été supprimé.

Exercice 4

Dans cet exercice, vous allez créer des blocs PL/SQL comportant des boucles et des structures de contrôle conditionnelles. vous allez :

- exécuter des actions conditionnelles en utilisant l'instruction IF
- écrire des schémas itératifs en utilisant la structure de la boucle

1. Exécutez la commande du fichier lab04_1.sql afin de créer la table MESSAGES. Ecrivez un bloc PL/SQL pour insérer des nombres dans la table MESSAGES.
a. Insérez les nombres de 1 à 10, en excluant 6 et 8.
b. Effectuez une validation (commit) avant la fin du bloc.
c. Affichez le contenu de la table MESSAGES pour tester votre bloc PL/SQL.

2. Créez un bloc PL/SQL qui calcule le montant de la commission d'un employé donné, en fonction de son salaire.
a. Utilisez la commande DEFINE pour fournir le numéro d'employé. Transmettez cette valeur au bloc PL/SQL à l'aide d'une variable de substitution.
DEFINE p_empno = 100
b. Si le salaire de l'employé est inférieur à 5 000 $, affichez le montant de la prime de l'employé, soit 10 % du salaire.
c. Si le salaire de l'employé est compris entre 5 000 $ et 10 000 $, affichez le montant de la prime de l'employé, soit 15 % du salaire.
d. Si le salaire de l'employé est supérieur à 10 000 $, affichez le montant de la prime de l'employé, soit 20 % du salaire.
e. Si le salaire de l'employé a la valeur NULL, affichez la valeur 0 en tant que montant de la prime de l'employé.
f. Testez le bloc PL/SQL pour chaque cas à l'aide du tableau suivant, et vérifiez le montant de chaque prime.
Remarque : Placez la commande SET VERIFY OFF avant le bloc PL/SQL.

Pour aller plus loin :

3. Créez une table `EMP` constituant une copie de la table `EMPLOYEES`. Pour cela, vous pouvez exécuter le script `lab04_3.sql`. Ajoutez à la table `EMP` une nouvelle colonne, `STARS`, de type `VARCHAR2` et de longueur 50, pour le stockage des astérisques (*).

4. Créez un bloc PL/SQL qui récompense les employés en leur attribuant un astérisque dans la colonne `STARS`, par tranche de salaire de 1 000 $.

a. Utilisez la commande `DEFINE` pour indiquer le numéro d'employé. Transmettez cette valeur au bloc PL/SQL à l'aide d'une variable de substitution.

```
DEFINE p_empno=104
```

b. Affectez à la variable `v_asterisk` la valeur initiale `NULL`.

c. Ajoutez un astérisque à la chaîne par tranche de salaire de 1 000 $. Par exemple, si l'employé possède un salaire de 8 000 $, la chaîne comportera 8 astérisques. Si l'employé possède un salaire de 12 500 $, la chaîne comportera 13 astérisques.

d. Mettez à jour la colonne `STARS` de l'employé avec la chaîne d'étoiles correspondante.

e. Effectuez la validation.

f. Testez le bloc en utilisant les valeurs suivantes :

```
DEFINE p_empno=174
DEFINE p_empno=176
```

g. Affichez les lignes de la table `EMP` pour vérifier si votre bloc PL/SQL a été correctement exécuté.

Remarque : Placez la commande `SET VERIFY OFF` avant le bloc PL/SQL.