

Oracle Database 11g : Les fondamentaux du langage PL/SQL

Présentation des exercices

Dans ces exercices, vous allez :

- + utiliser **Oracle SQL Developer** pour examiner des objets de données dans le compte **HR** (déverrouillé préalablement).
- + Tous les exercices écrits utilisent l'environnement de développement Oracle SQL Developer. Vous pouvez aussi utiliser **SQL*Plus**.

Avant de commencer les exercices, notez les points suivants :

- + Enregistrez tous vos fichiers d'exercices dans des scripts nommés **lab_<numero>.sql** dans un répertoire créé préalablement : **c:\EXPL**
- + Entrez vos instructions SQL dans une feuille de calcul SQL Worksheet. Pour enregistrer un script dans SQL Developer, vérifiez que la feuille de calcul SQL requise est active puis sélectionnez **Fichier→Enregistrer sous**. Vous pouvez aussi cliquer avec le bouton droit de la souris dans la feuille de calcul SQL Worksheet et sélectionner **Save file** pour enregistrer votre instruction SQL dans d'un script nommé **lab<numero>.sql**
- + Lorsque vous modifiez un script existant, veillez à utiliser l'option **Save As** pour l'enregistrer sous un autre nom.
- + Pour exécuter l'interrogation, cliquez sur l'icône **Exécuter l'instruction** dans la feuille de calcul SQL Worksheet. Vous pouvez également appuyer sur **[F9]**.
- + Pour les instructions **DML** et **DDL**, utilisez l'icône **Exécuter un Script** ou appuyez sur **[F5]**.
- + Une fois l'interrogation exécutée, veillez à ne pas entrer l'interrogation suivante dans la même feuille de calcul. Ouvrez une nouvelle feuille.

IMPORTANT

Les TP sont à rendre en groupe sur Moodle. Le délai est de 7 jours, tous les membres d'un même groupe doivent valider la remise du travail.

1. Exercice 1 (1 pt)

1. Parmi les blocs PL/SQL suivants, lesquels s'exécutent avec succès ?

```
a.  
BEGIN  
END;  
  
b.  
DECLARE  
amount INTEGER(10);  
END;  
  
c.  
DECLARE  
BEGIN  
END;  
  
d.  
DECLARE  
amount INTEGER(10);  
BEGIN  
DBMS_OUTPUT.PUT_LINE(amount);  
END;
```

2. Créez et exécutez un bloc anonyme simple. Exécutez ce script et enregistrez-le sous le nom **lab_01_sol.sql**

2. Exercice 2 (1 pt)

1. Déterminez les noms d'identificateur valides et non valides :

- a. today
- b. last_name
- c. today's_date
- d. Number_of_days_in_February_this_year
- e. Isleap\$year
- f. #number
- g. NUMBER#
- h. number1to7

2. Identifiez les déclarations et initialisations valides et non valides des variables :

- a. number_of_copies PLS_INTEGER;
- b. printer_name constant VARCHAR2(10);
- c. deliver_to VARCHAR2(10):=Johnson;
- d. by_when DATE:= CURRENT_DATE+1;

3. Examinez le bloc anonyme suivant et choisissez l'affirmation appropriée.

```
SET SERVEROUTPUT ON
DECLARE
v_fname VARCHAR2(20);
v_lname VARCHAR2(15) DEFAULT 'fernandez';
BEGIN
DBMS_OUTPUT.PUT_LINE(v_fname || ' ' || v_lname);
END;
/
```

- a. Le bloc s'exécute avec succès et affiche "fernandez".
- b. Le bloc renvoie une erreur, car la variable fname est utilisée sans initialisation.
- c. Le bloc s'exécute avec succès et affiche "null fernandez".
- d. Le bloc renvoie une erreur, car vous ne pouvez pas utiliser le mot-clé DEFAULT pour initialiser une variable de type VARCHAR2.
- e. Le bloc génère une erreur, car la variable v_fname n'est pas déclarée.

4. Créez un bloc anonyme. Dans SQL Developer, chargez le script lab_01_sol.sql que vous avez créé à la question 2 de l'exercice 1.

a. Ajoutez, si elle est absente, une section déclarative à ce bloc PL/SQL. Dans cette section, déclarez les variables suivantes :

- 1. Variable **v_today** de type **DATE**. Initialisez **today** avec **SYSDATE**.
- 2. Variable **v_tomorrow** de type **today**. Utilisez l'attribut **%TYPE** pour déclarer cette variable.

b. Dans la section exécutable, initialisez la variable **tomorrow** avec une expression qui calcule la date du lendemain (ajoutez la valeur "1" à la valeur de **today**). Affichez la valeur de **today** et de **tomorrow**.

c. Exécutez ce script et enregistrez-le sous le nom **lab_02_sol.sql**.

3. Exercice 3 (1 pt)

1. 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;

        1|
    END;

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

    2|
END;
/
```

Examiner le bloc PL/SQL qui précède et déterminez le type de données et la valeur de chacune des variables suivantes d'après les règles relatives à la 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 :

2. Exemple relatif à 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_credit_rating := 'GOOD';
        ...
    END;
...
END;
/
```

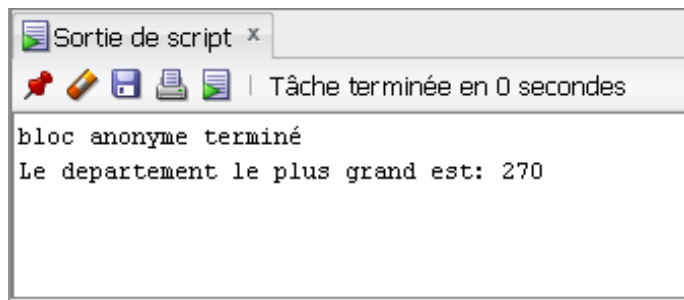
Dans le bloc PL/SQL qui précède, déterminez les valeurs et les types de données pour chacun des cas suivants.

- La valeur de **v_customer** dans le bloc imbriqué est :
- La valeur de **name** dans le bloc imbriqué est :
- La valeur de **v_credit_rating** dans le bloc imbriqué est :
- La valeur de **v_customer** dans le bloc principal est :
- La valeur de **name** dans le bloc principal est :
- La valeur de **v_credit_rating** dans le bloc principal est :

4. Exercice 4 (1 pt)

1. Créez un bloc PL/SQL qui sélectionne l'ID de département le plus grand dans la table **departments** et qui le stocke dans la variable **v_max_deptno**. Affichez l'ID de département le plus élevé.

- Déclarez une variable **v_max_deptno** de type **NUMBER** dans la section déclarative.
- Commencez la section exécutable par le mot-clé **BEGIN** et incluez une instruction **SELECT** afin d'extraire la valeur **department_id** maximale de la table **departments**.
- Affichez **v_max_deptno** et terminez le bloc exécutable.
- Exécutez votre script et enregistrez-le sous le nom **lab_04_sol.sql**.

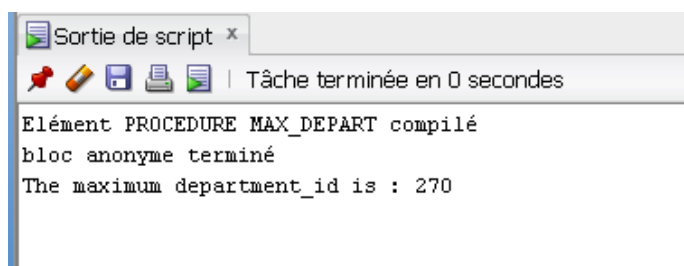


```
Sortie de script x
Tâche terminée en 0 secondes

bloc anonyme terminé
Le departement le plus grand est: 270
```

5. Exercice 5 (1 pt)

1. Transformer le bloc anonyme de l'exercice 4 en une procédure stockée **MAX_DEPART**.
 - a. Remplacer l'entête du bloc par « **CREATE OR REPLACE PROCEDURE MAX_DEPART AS** ».
 - b. Compiler la procédure dans la base de données.
 - c. Sauvegarder la procédure dans le fichier lab_05_sol.sql. Exécutez avec un bloc PL/SQL anonyme et/ou avec la commande « *execute* » :

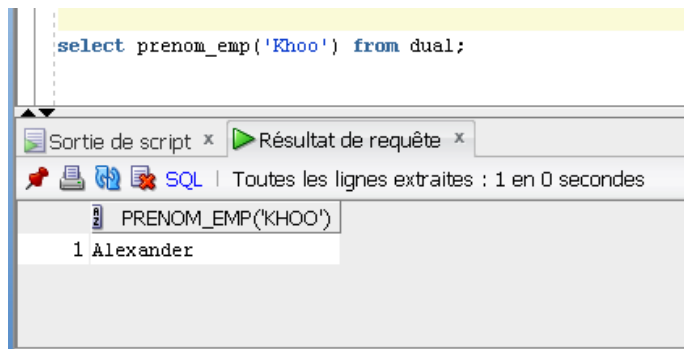


```
Sortie de script x
Tâche terminée en 0 secondes

Elément PROCEDURE MAX_DEPART compilé
bloc anonyme terminé
The maximum department_id is : 270
```

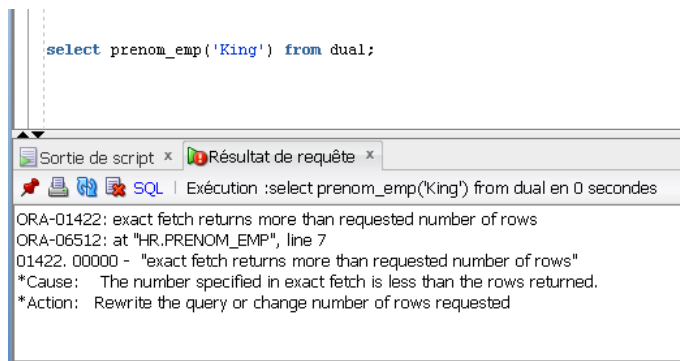
6. Exercice 6 (2 pt)

1. Créer une fonction **PRENOM_EMP** qui retourne le prénom d'un employé en prenant comme argument paramètre le nom (Last_name) de la personne.
 - a. déclarer le paramètre de la fonction en utilisant l'attribut **%TYPE** -> **nom_fonction(nom_paramètre table.colonne%TYPE)**.
 - b. Sauvegarder la fonction dans le fichier lab_06_sol.sql et la compiler dans la base de données.
 - c. Tester la fonction avec un ordre SELECT :



7. Exercice 7 (2 pt)

1. Lorsque la fonction de l'exercice 6 retourne plus d'une ligne une erreur est affichée.

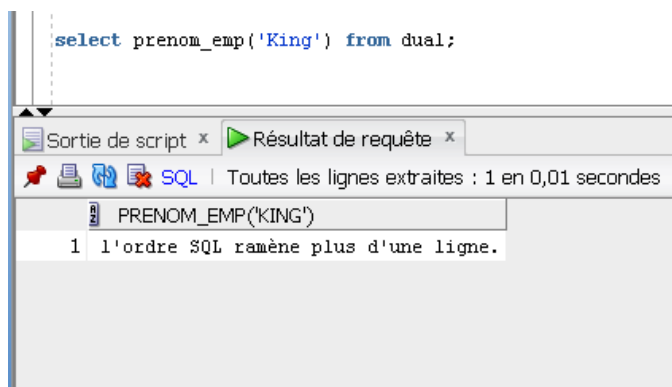


Modifier la fonction pour prendre en compte l'erreur **1422**

a. Ajouter une section **EXCEPTION** avec l'exception pré-définie **TOO_MANY_ROWS**. Retourner le message « L'Ordre SQL ramène plus d'une ligne ». Indication : pour afficher une cote « ' » il faut doubler le caractère -> 'L'ordre ... d'une...'

b. Sauvegarder la fonction dans le fichier lab_07_sol.sql et la compiler dans la base de données.

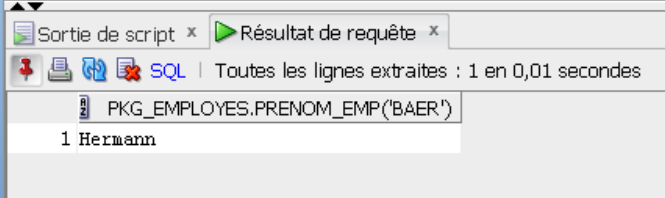
c. Tester la fonction avec un ordre SELECT qui ramène plusieurs lignes :



8. Exercice 8 (1 pt)

1. Créez le package **PKG_EMPLOYES** pour contenir les procédures et fonctions que vous venez de créer. Tester les procédures et fonctions packagées.
 - a. Créer la partie spécification ainsi que le corps du « package ». Mettre des commentaires et documenter les fonctionnalités
 - b. Sauvegarder le package dans le fichier lab_08_sol.sql et le compiler dans la base de données.
 - c. Tester les procédures et fonctions.

```
select PKG_EMPLOYES.prenom_emp('Baer') from dual;  
exec PKG_EMPLOYES.max_depart;
```

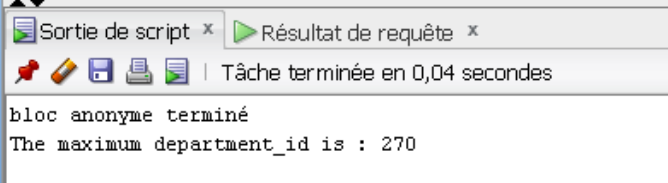


Sortie de script x Résultat de requête x

Toutes les lignes extraites : 1 en 0,01 secondes

PKG_EMPLOYES.PRENOM_EMP('BAER')
1 Hermann

```
exec PKG_EMPLOYES.max_depart;
```



Sortie de script x Résultat de requête x

Tâche terminée en 0,04 secondes

bloc anonyme terminé
The maximum department_id is : 270