**PL/SQL : Exemples & Exercices**

**Section Déclarative :**

**Exemple 1 :**

Tester les déclarations suivantes :

**DECLARE**

**V\_x NUMBER NOT NULL := 10 ;**

**V\_hiredate DATE NOT NULL ;**

**V\_nb INTEGER DEFAULT 100 ;**

**V\_chaine1, v\_chaine2 VARCHAR2(30) ;**

**V\_var BOOLEAN := 1 ;**

**BEGIN**

**NULL ;**

**END ;**

**Exemple 2 :**

Découvrir les blocs internes et les procédures prédéfinies.

**Set SERVEROUTPUT ON**

**DECLARE**

**V\_x VARCHAR2(50) ;**

**V\_y VARCHAR2(50) ;**

**PROCEDURE Proc1 (str IN VARCHAR2) IS**

**BEGIN**

**DBMS\_OUTPUT.PUT\_LINE(str) ;**

**END Proc1 ;**

**BEGIN**

**V\_x  := ‘chaine1’ ; v\_y :=’chaine2’ ;**

**IF (v\_x=v\_y) THEN**

**Proc1(‘Les chaines sont Egales !’) ;**

**ELSE**

**Proc1(‘Les chaines ne sont pas Egales !!’) ;**

**END IF ;**

**Proc1( LENGTH(v\_x)) ;**

**Proc1( LENGTH(v\_y)) ;**

**END ;**

**/**

**Exemple 3 :**

**Variables PLSQL / BIND / Substitution**Découvrir les différences entre la déclaration et l’utilisation des variables PLSQL et celles des variables NON PLSQL.

**VARIABLE g\_chaine VARCHAR2(50)**

**VARIABLE g\_resultat NUMBER**

**ACCEPT g\_nombre1 PROMPT ‘Entrez le nombre 2 : ‘**

**DECLARE**

**V\_nombre2 NUMBER NOT NULL := 10 ;**

**BEGIN**

**IF (&g\_nombre1 <> 0) THEN**

**:g\_resultat := (v\_nombre2 + 12)/ &g\_nombre1 ;**

**:g\_chaine :=’la division est terminée avec succès !’ ;**

**ELSE**

**:g\_chaine :=’Erreur : la division sur 0 !’ ;**

**END ;**

**/**

**PRINT g\_chaine**

**PRINT g\_resultat**

**Les variables Hôtes (Bind Variable)**

**Les variables de substitution**

**L’utilisation des variables Hôtes avec ‘ : ’ et les variables de substitution avec ‘ & ‘**

**Affichage des variables Hôtes avec « PRINT »**

**Exercice :**

Codez un bloc PL/SQL qui calcule le gain total pour une année. Le salaire annuel et le pourcentage de bonus de l’année sont transmis au bloc PL/SQL à l’aide de variables de substitution SQL\*PLUS, et le bonus doit être converti d’un nombre entier en décimal (par exemple, 15 en 0,15). Si le salaire est NULL, définissez-le à zéro avant de calculer le traitement total. Exécutez le bloc PL/SQL.

Exemple :  
Veuillez saisir le salaire annuel : 50000

Veuillez saisir le % de bonus : 10

TOTAL

----------

55000

**Section Exécutable :**

**Exemple 1 : la porté des variables**

Remplir le tableau au-dessous en se basant sur le bloc PL/SQL suivant :

|  |  |  |
| --- | --- | --- |
| **Variable** | **Valeur dans le bloc Principal** | **Valeur dans le sous-bloc** |
| **V\_nombre** |  |  |
| **V\_message** |  |  |
| **V\_test** |  |  |
| **V\_local** |  |  |

**SET SERVEROUTPUT ON**

**DECLARE**

**v\_nombre NUMBER := 600;**

**v\_message VARCHAR2(100):=‘Message Principal’;**

**v\_test VARCHAR2(10) := ‘True’;**

**BEGIN**

**DECLARE**

**v\_nombre NUMBER :=1;**

**v\_message VARCHAR2(100):=‘Message second’;**

**v\_local VARCHAR2(100):=‘Message Interne’;**

**BEGIN**

**v\_nombre:=v\_nombre+1;**

**v\_local:=‘MSG1: ‘||v\_local;**

**DBMS\_OUTPUT.PUT\_LINE(‘Les valeurs des variables du sous-bloc :’) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_nombre= ’||to\_char(v\_nombre)) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_message= ’||v\_message) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_local= ’||v\_local) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_test= ’||v\_test) ;**

**END;**

**v\_nombre:=v\_nombre+1;**

**v\_message:=v\_message||’MSG2’;**

**v\_local:=‘MSG3 : ’||v\_local;**

**DBMS\_OUTPUT.PUT\_LINE(‘Les valeurs des variables du bloc principal :’) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_nombre= ’||to\_char(v\_nombre)) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_message= ’||v\_message) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_test= ’||v\_test) ;**

**DBMS\_OUTPUT.PUT\_LINE(‘v\_local= ’||v\_local) ;**

**END;**

**Exemple 2 :**

**Instructions PL/SQL**

**ACCEPT g\_employe PROMPT ‘Entrez le numéro de l’employé : ‘**

**VARIABLE g\_message VARCHAR2(60)**

**DECLARE**

**V\_sal emp.sal%TYPE ;**

**V\_comm emp.comm%TYPE ;**

**BEGIN**

**Select sal, comm into v\_sal, v\_comm**

**From emp where empno=&g\_employe ;**

**IF (NVL(v\_sal, 0) >1500) THEN**

**V\_comm :=NVL(v\_sal,0)\*0.2 ;**

**ELSIF (NVL(v\_sal, 0) <1000) THEN**

**V\_comm :=NVL(v\_sal,0)\*0.1;**

**ELSE**

**V\_comm :=NVL(v\_sal,0)\*0.15 ;**

**END IF ;**

**UPDATE emp set comm=v\_comm**

**Where empno=&g\_employe ;**

**COMMIT ;**

**:g\_message :=’La commission est mise à jour par la valeur : ‘||to\_char(v\_comm) ;**

**SAVEPOINT save1 ;**

**INSERT INTO emp(empno, ename) VALUES(8000, ‘MASTER ‘) ;**

**SAVEPOINT save2 ;**

**INSERT INTO emp(empno, ename) VALUES(9000, ‘SID‘) ;**

**DELETE FROM emp where empno=9000 ;**

**ROLLBACK to SAVEPOINT save2;**

**END ;**

**/**

**PRINT g\_message**

**Select empno, ename, sal, comm from emp ;**

**Exemple 3 :**

Ajouter une nouvelle colonne STARS varchar2(50), dans la table EMP qui permettra de stocker des étoiles (\*).

Créer un programme PL/SQL qui récompense les employés en leur attribuant une étoile dans la colonne STARTS par tranche de salaire de $100.

**Alter table emp add STARS carchar2(100) ;**

**ACCEPT g\_num\_emp PROMPT ‘Entrez le numéro de l’employé : ‘**

**DECLARE**

**V\_star VARCHAR2(100) :=’\*’ ;**

**V\_sal emp.sal%TYPE ;**

**V\_cpt NUMBER :=0 ;**

**BEGIN**

**Select sal into v\_sal from emp**

**Where empno=&g\_num\_emp ;**

**WHILE (v\_cpt< ROUND(v\_sal/100))**

**LOOP**

**V\_star :=v\_star||’\*’ ;**

**V\_cpt :=v\_cpt+1 ;**

**END LOOP ;**

**UPDATE emp SET STARS=v\_star**

**Where empno=&g\_num\_emp ;**

**COMMIT ;**

**END ;**

**/**

**Select ename, sal, stars from emp ;**

**Exemple 4 : Curseurs SQL**

**DECLARE**

**--declaration du curseur**

**CURSOR c\_deptEmp IS**

**--l’instruction select**

**select ename, job, dname from emp e, dept d**

**where e.deptno=d.deptno ;**

**--variables d’acceuil**

**v\_ename emp.ename%TYPE;**

**v\_job emp.job%TYPE;**

**v\_dname dept.dname%TYPE;**

**BEGIN**

**--ouverture du curseur**

**open c\_deptEmp;**

**--boucle sur les lignes**

**loop**

**--lecture d’une ligne**

**fetch c\_deptEmp into v\_ename, v\_job, v\_dname;**

**dbms\_output.put\_line(‘l’employé ‘||v\_ename||’ travaille comme ‘||v\_job||’ dans le departement ‘||v\_dname) ;**

**--sortir lorsque le curseur ne ramène pas de ligne**

**EXIT When c\_deptEMP%NOTFOUND;**

**end loop;**

**--fermeture du curseur**

**close c\_deptEMP;**

**END ;**

**/**

**Exercice 1 :**

Créer une table Messages ayant un RESULT de type chaine de caractères.

Ecrire un bloc PL/SQL pour insérer les nombres de 1 à 10, en excluant 6 et 8.

Afficher le contenu de la table Messages.

Ecrire le code nécessaire pour afficher le texte « Nombre pair » ou « Nombre impair », selon que le nombre correspondant dans la table Messages est pair ou impair.

**--création de la table Messages**

**CREATE TABLE Messages (Result VARCHAR2(5)) ;**

**--Remplissage de la table Messages**

**BEGIN**

**FOR v\_i IN 1..10 LOOP**

**IF (v\_i <> 6) AND (v\_i <> 8) THEN**

**INSERT INTO Messages VALUES (v\_i) ;**

**END IF ;**

**END LOOP ;**

**COMMIT ;**

**END ;**

**/**

**--Affichage de la table Messages**

**Select \* from Messages ;**

**--Impair/Pair**

**Set SERVEROUTPUT ON**

**DECLARE**

**CURSOR c\_message IS select \* from Messages ;**

**V\_num Messages.result%TYPE ;**

**V\_rest NUMBER :=0;**

**BEGIN**

**Open c\_message ;**

**LOOP**

**Fetch c\_message into v\_num ;**

**EXIT When c\_message%NOTFOUND ;**

**V\_rest :=TO\_NUMBER(v\_num);**

**IF (MOD(v\_rest ,2)<>0) THEN**

**DBMS\_OUTPUT.PUT\_LINE(v\_num||’ ---> Nombre impair’) ;**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE(v\_num||’ ---> Nombre pair’) ;**

**END IF ;**

**END LOOP ;**

**Close c\_message ;**

**END ;**

**/**

**Exercice 2 :**

On souhaite appliquer une règle d’échantillonnage à la table EMP(empno, ename,…).

Ecrire un programme PLSQL permettant de lire la table EMP et d’afficher les noms des employés dont les rangs d’apparition dans la table sont : 1, 3, 6, 10, 15,….

Règles de calcul des rangs successifs :

1=0+1

3=1+2

6=3+3

10=6+4

15=10+5

Résultat attendu :

1 : SMITH

3 : BLAKE

6 : WARD

10 : TURNER

**Bon Apprentissage !**