

Cours	Programmation PL/SQL
Auditoire	L_BD_2 et L_GLSI_2
Établissement	UC
Responsable du cours	Marwa CHAIEB
Années Universitaires	2020/2021 (UC)

Description du cours

Ce cours permet la compréhension de la programmation PL/SQL à travers la présentation des concepts de base utilisés.

Chaque séance traite un point important illustré par des exemples, exercices et travaux pratiques.

A la fin du cours, l'étudiant sera capable de comprendre les spécificités des différents éléments.

Objectifs du cours :

À l'issue de ce cours, l'étudiant(e) doit être capable de :

- Présenter la structure d'un bloc PL/SQL
- Définir la notion de variable et présenter les différents types
- Utiliser les structures de contrôles
- Gérer les erreurs par des exceptions
- Définir des déclencheurs
- Définir des fonctions et des procédures

Langue d'enseignement :

☐

Arabe

☒

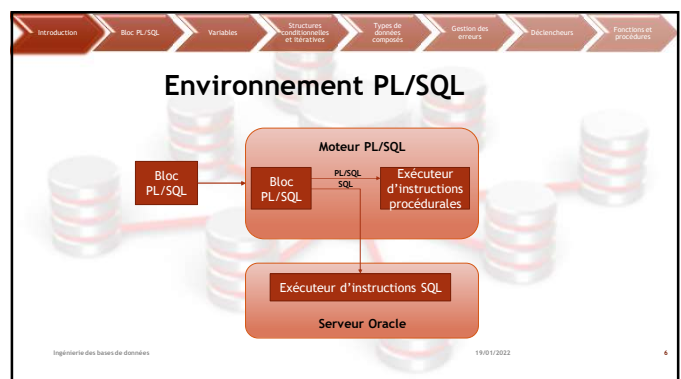
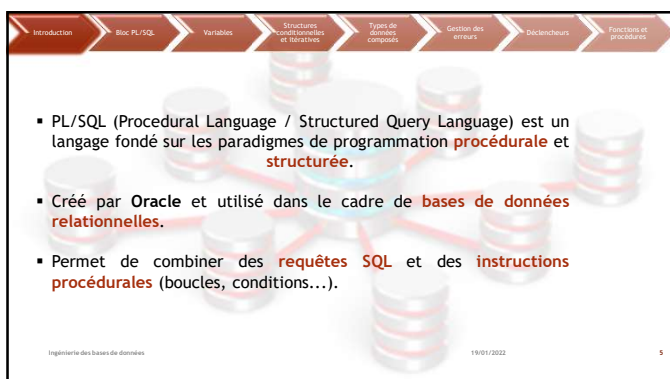
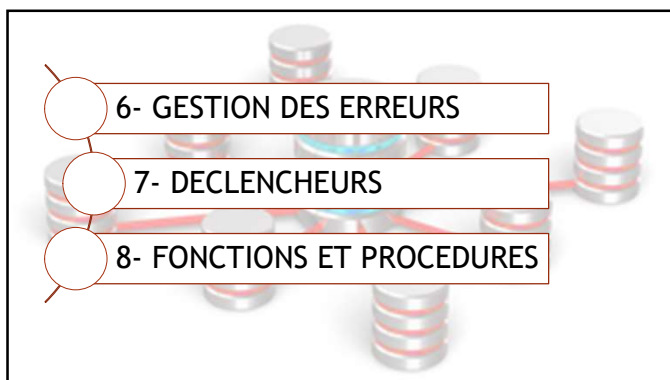
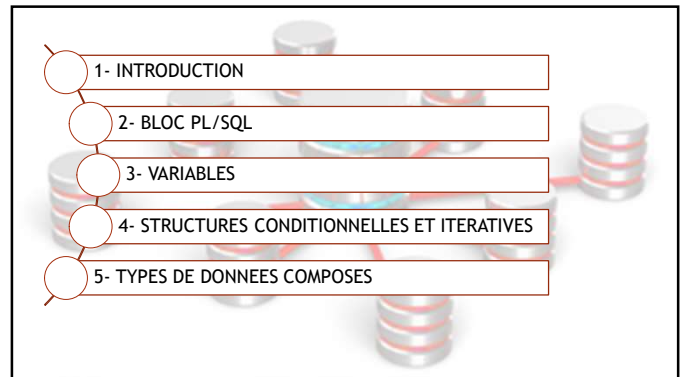
Français

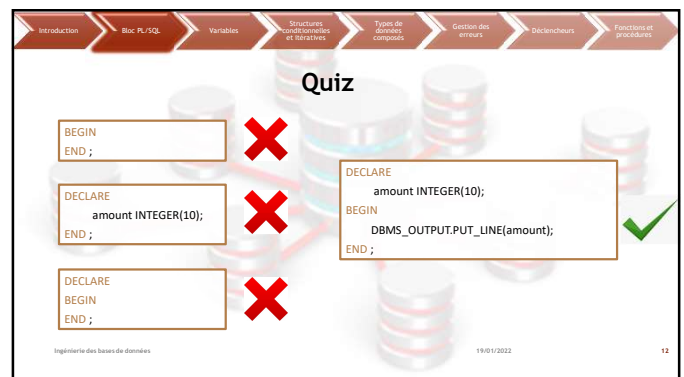
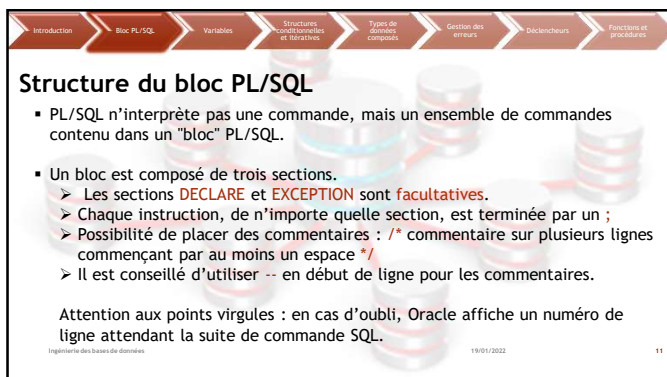
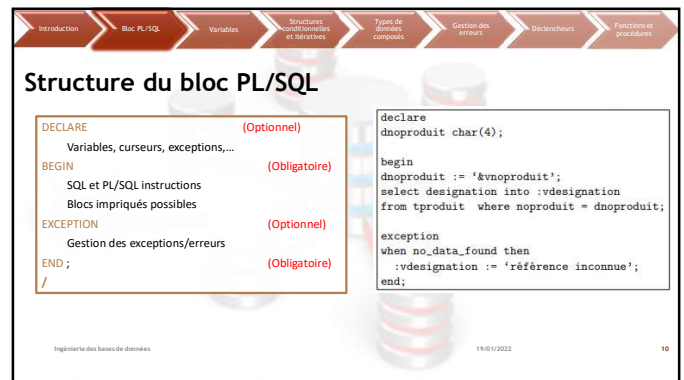
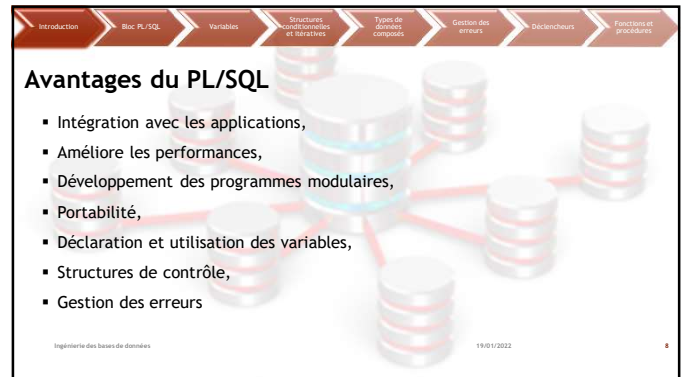
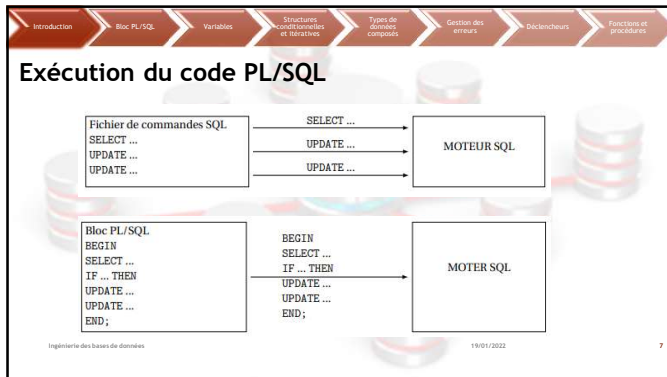
☐

Anglais

Déroulement du cours

Semaine	Contenu	Références	Objectifs spécifiques
1	Présentation du cours Chapitre 1 : INTRODUCTION	(1) P5-P8	Définition de la programmation PL/SQL
2	Chapitre 2 : STRUCTURE D'UN BLOC PL/SQL	(2) 9-14	Présenter la structure et les types du bloc PL/SQL
3	Chapitre 3 : NOTION DES VARIABLES	(3) 15- 33	Maitriser la notion des variables et leurs différents types
4	Chapitre 4 : STRUCTURES CONDITIONNELLES	(4) 34-40	Présenter les différentes structures conditionnelles
5	Chapitre 4 (suite) : STRUCTURES ITÉRATIVES	(5) 41-48	Présenter les différentes boucles
6	TP1 : STRUCTURES DE CONTROLE		
	Chapitre 5 : GESTION DES ERREURS	(6) 49-65	Maitriser la notion d'exception et ses différents catégories
7	Chapitre 5 (suite)		
8	Devoir surveillé		
9	Chapitre 6 : DÉCLENCHEURS	(7) 66-83	Comprendre la notion et l'utilité des déclencheurs
10	EXERCICES		
11	Chapitre 7 : FONCTIONS ET PROCÉDURES	(8) 84-97	Maitriser la notion des sous-programmes
12	EXERCICES		
13	Semaine de préparation aux examens	Sujets des années antérieures	Être capable de résoudre les problèmes soulevés par ces sujets
14	Examen Final		





Types de blocs PL/SQL

```

Anonyme
[DECLARE]
BEGIN
  Instructions...
[EXCEPTION]
END ;
/

Sous-programmes
Procédure
PROCEDURE nom
IS
BEGIN
  Instructions...
[EXCEPTION]
END ;
/

Fonction
FUNCTION nom
RETURN type de retour
IS
BEGIN
  Instructions...
  RETURN valeur;
[EXCEPTION]
END ;
/
  
```

Ingénierie des bases de données 19/01/2022 13

Blocs anonymes vs Sous-programmes

- Bloc sans nom
- Compilé chaque fois
- Non sauvegardé dans la base de donnée
- Ne peut pas être appelé par d'autres applications
- Ne retourne pas de valeurs
- Ne prend pas des paramètres

- Bloc nommé
- Compilé une seule fois
- Sauvegardé dans la base de donnée
- Peut être appelé par d'autres applications
- Les fonctions retournent pas de valeurs
- Peut prendre des paramètres

Ingénierie des bases de données 19/01/2022 14

CH 3
Les Variables

Utilisation des variables

- Les variables sont utilisées pour:
 - Stocker temporairement des données,
 - Manipuler les données sauvegardées,
 - Réutilisation des données.

Ingénierie des bases de données 19/01/2022 16

Conditions sur les noms des variables

- Le nom d'une variable:
 - Doit commencer par une lettre,
 - Peut contenir des lettres, des chiffres et des caractères spéciaux(\$, _, #),
 - Doit contenir, au maximum, 30 caractères,
 - Ne doit pas contenir des mots clés.

Ingénierie des bases de données 19/01/2022 17

Déclaration d'une variable

- La partie déclarative contient la déclaration de toutes les variables locales utilisées dans la partie exécutable, ainsi que leurs initialisations.

```

DECLARE
  ch VARCHAR2(15) := 'Hello World';
BEGIN
  DBMS_OUTPUT.PUT_LINE( ch );
END ;
/
  
```

Ingénierie des bases de données 19/01/2022 18

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Déclaration d'une variable/constante

- Syntaxe
 - nom [CONSTANT] type [NOT NULL] [: = Expression];
- Exemple


```
DECLARE
  v_dateOfBirth DATE;
  v_dep Number(2) NOT NULL := 10;
  v_city VARCHAR2(13) := 'Tunis';
  c_cumul CONSTANT NUMBER := 1000;
```

Ingénierie des bases de données 19/01/2022 19

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Déclaration d'une variable/constante

- Nom** : Le nom de la variable,
- CONSTANT** : Indique que la valeur ne peut pas être modifiée, dans ce cas **expression** doit être indiquée,
- NOT NULL** : Indique que la variable ne peut pas être nulle, dans ce cas **expression** doit être indiquée,
- type** : Représente le type de la variable.

Ingénierie des bases de données 19/01/2022 20

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Types des variables

- VARCHAR2(n)**: Chaîne de caractère de taille maximale n
- NUMBER(n)**: Nombre de taille n
- DATE**: Une date
- CHAR(n)**: Chaîne de caractère de taille n
- BOOLEAN**: 0 ou 1
- INTEGER**: entier.

Ingénierie des bases de données 19/01/2022 21

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Exemples

```
v_nom t.nom% TYPE; --t est le nom d'une table
v_salaire_annuel NUMBER (7,2);
v_salaire_mensuel v_salaire_annuel % TYPE := 2000;
```

Ingénierie des bases de données 19/01/2022 22

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Variable globale

- Pour déclarer une variable globale, elle doit être précédée par référence ': '.
- Exemples:
 - Stocker le salaire mensuel dans une variable globale:


```
:g_salaire_mensuel := v_salaire_annuel / 12;
```
 - Utilisation du salaire mensuel dans un bloc PL/SQL:


```
IF :g_salaire_mensuel > 1200 THEN ....
```

Ingénierie des bases de données 19/01/2022 23

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Exercice 1.

Écrire un fichier SQL qui permet d'afficher le message "Entrer votre age :", de saisir l'âge de l'utilisateur dans la variable v_age, d'ajouter un à cette valeur et d'afficher "Votre age plus un est 20 ans" si 19 est la valeur saisie par l'utilisateur et 20 le contenu de la variable v_age.

Ingénierie des bases de données 19/01/2022 24

Exercice 2.

Parmi les déclarations de variables suivantes, déterminer celles qui sont incorrectes :

A - DECLARE
v_id NUMBER(4); ✓

B - DECLARE
v_x,v_y,v_z VARCHAR2(10); ✗

C - DECLARE
v_date_naissance DATE NOT NULL; ✗

D - DECLARE
v_en_stock BOOLEAN := 1; ✗

E - DECLARE
emp_record emp_record_type; ✗

Ingénierie des bases de données 19/01/2022 25

Affectation d'une valeur

- L'affectation d'une valeur dans une variable se fait avec :=
- Elle peut se faire aussi avec SELECT ... INTO:

```
SELECT select_list
INTO var_name[,var_name2...]
FROM table
WHERE condition;
```

- La requête doit retourner une seule valeur par colonne.

Ingénierie des bases de données 19/01/2022 26

Exemple 1

```
1 DECLARE
2   v_customer_name customers.name%TYPE;
3 BEGIN
4   SELECT
5     name
6   INTO
7     v_customer_name
8   FROM
9     customers
10  WHERE
11    customer_id = 100;
12
13 DBMS_OUTPUT.PUT_LINE( v_customer_name );
14 END;
```

Ingénierie des bases de données 19/01/2022 27

Exemple 2

```
DECLARE
  v_deptnum NUMBER(2);
  v_loc VARCHAR2(15);
BEGIN
  SELECT deptnum, loc
  INTO v_deptnum, v_loc
  FROM dept
  WHERE name_dep = 'INFO';
END ;
/
```

Ingénierie des bases de données 19/01/2022 28

Exemple 3

```
DECLARE
  v_sum_sal emp.salary%TYPE;
  v_deptnum NOT NULL NUMBER := 10;
BEGIN
  SELECT SUM(salary)
  INTO v_sum_sal
  FROM emp
  WHERE deptnum = v_deptnum;
END ;
/
```

Ingénierie des bases de données 19/01/2022 29

Utilisation des variables

- Exemple 1:

```
DECLARE
  v_empnum NOT NULL NUMBER := 105;
BEGIN
  INSERT INTO emp
  VALUES (v_empnum, 'Clement', 'Manager', 10);
END ;
/
```

Ingénierie des bases de données 19/01/2022 30

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Utilisation des variables

Exemple 2:

```
DECLARE
    v_augm emp.sal%TYPE := 2000;
BEGIN
    UPDATE emp
    SET sal := sal + v_augm
    WHERE job = 'Lecturer';
END ;
/
```

Ingénierie des bases de données 19/01/2022 31

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Utilisation des variables

Exemple 3:

```
DECLARE
    v_deptnum emp.deptnum%TYPE := 10;
BEGIN
    DELETE FROM emp
    WHERE deptnum = v_deptnum;
END ;
/
```

Ingénierie des bases de données 19/01/2022 32

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Types de données composés Gestion des erreurs Déclencheurs Fonctions et procédures

Exercices

Ingénierie des bases de données 19/01/2022 33



CH 4
Les structures de contrôle

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les structures de contrôle en PL/SQL

- La structure conditionnelle IF:
 - IF THEN END IF;
 - IF THEN ELSE END IF;
 - IF THEN ELSEIF THEN END IF;
- Les boucles:
 - LOOP END LOOP;
 - FOR LOOP END LOOP;
 - WHILE LOOP END LOOP;

Ingénierie des bases de données 19/01/2022 35

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

La structure conditionnelle

```
IF condition THEN
    instructions;
[ELSIF condition THEN
    instructions;]
[ELSE
    instructions;]
END IF;
```

```
IF v_name = 'Clement' THEN
    v_pos := 'Teacher';
    v_deptnum := 102;
    v_sal := sal * 0.25;
END IF;
```

Ingénierie des bases de données 19/01/2022 36

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

La structure conditionnelle

```

IF condition THEN
  instructions;
[ELSIF condition THEN
  instructions;]
[ELSE
  instructions;]
END IF;

IF v_name = 'Clement' THEN
  v_pos := 'Teacher';
  v_deptnum := 102;
  v_sal := sal * 0.25;
ELSE
  DBMS_OUTPUT.PUT_LINE('Non-existent employee');
END IF;

```

Ingenierie des bases de données 19/01/2022 37

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

La structure conditionnelle

```

IF condition THEN
  instructions;
[ELSIF condition THEN
  instructions;]
[ELSE
  instructions;]
END IF;

IF v_beginning = >100 THEN
  RETURN (2*v_beginning);
ELSIF v_beginning = >=50 THEN
  RETURN (5*v_beginning);
ELSE
  RETURN (v_beginning);
END IF;

```

Ingenierie des bases de données 19/01/2022 38

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

La structure conditionnelle

```

IF condition THEN
  instructions;
[ELSIF condition THEN
  instructions;]
[ELSE
  instructions;]
END IF;

IF v_beginning = >100 THEN
  RETURN (2*v_beginning);
ELSIF v_beginning = >=50 THEN
  RETURN (5*v_beginning);
ELSE
  RETURN (v_beginning);
END IF;

```

Ingenierie des bases de données 19/01/2022 39

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exercices

Ingenierie des bases de données 19/01/2022 40

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les structures itératives

Loop basique:

```

LOOP
  instructions;
...
EXIT [WHEN condition];
END LOOP;

DECLARE
  v_date DATE;
  v_count NUMBER(2) := 1;
BEGIN
  v_date := SYSDATE;
  LOOP
    INSERT INTO article
      VALUES (v_count,v_date);
    v_count := v_count +1;
    EXIT WHEN v_count >10;
  END LOOP;
END ;
/

```

Ingenierie des bases de données 19/01/2022 41

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les structures itératives

Loop basique:

```

LOOP
  instructions;
...
EXIT [WHEN condition];
END LOOP;

DECLARE
  v_date DATE;
  v_count NUMBER(2) := 1;
BEGIN
  v_date := SYSDATE;
  LOOP
    INSERT INTO article
      VALUES (v_count,v_date);
    v_count := v_count +1;
    EXIT WHEN v_count >10;
  END LOOP;
END ;
/

```

Ingenierie des bases de données 19/01/2022 42

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les structures itératives

Boucle For:

```
FOR indice IN [REVERSE] inf...sup LOOP
  instructions;
...
END LOOP;
```

NB:

- On n'a pas besoin de déclarer l'indice, il est déclaré implicitement
- L'option **REVERSE** permet de parcourir en arrière

Ingénierie des bases de données 19/01/2022 43

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les structures itératives

Boucle For:

```
DECLARE
  v_date DATE;
BEGIN
  v_date := SYSDATE;
  FOR i IN 1..&Nb LOOP
    INSERT INTO article
      VALUES (i,v_date);
  END LOOP;
END ;
/
```

Avec &Nb, le système attend une valeur de l'utilisateur au début de la boucle for.

Ingénierie des bases de données 19/01/2022 44

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les structures itératives

Boucle While:

```
WHILE condition LOOP
  instructions;
...
END LOOP;
```

```
DECLARE
  v_date DATE;
  v_count NUMBER(2) := 1;
BEGIN
  v_date := SYSDATE;
  WHILE v_count < 10 LOOP
    INSERT INTO article
      VALUES (v_count,v_date);
    v_count := v_count + 1;
  END LOOP;
END ;
/
```

NB:

- La condition est évaluée avant chaque itération

Ingénierie des bases de données 19/01/2022 45

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les structures itératives

Boucles imbriquées et labels

- On peut imbriquer les boucles à plusieurs niveaux
- Utiliser des labels pour distinguer entre les blocs et les boucles
- Quitter les boucles avec EXIT faisant référence au label
- Les labels s'écrit: « nom_label »

```
DECLARE
  i number(1);
  j number(1);
BEGIN
  << outer_loop >>
  FOR i IN 1..3 LOOP
    << inner_loop >>
    FOR j IN 1..3 LOOP
      dbms_output.put_line('i is: '|| i || ' and j is: ' || j);
    END loop inner_loop;
  END loop outer_loop;
END;
```

Ingénierie des bases de données 19/01/2022 46

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exercices 1:

Calculer le factoriel d'un entier n saisi au clavier

Exercice 2:

Soit la table tab1, Écrivez un bloc PL/SQL qui insère les chiffres de 1 à 100 dans cette table.

Ingénierie des bases de données 19/01/2022 47

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exercice 3:

La déclaration d'un tableau se fait de la manière suivante:

```
TYPE MONTAB IS VARRAY (50) OF INTEGER ;
t MONTAB ;
```

- 1) Écrivez un bloc pl/sql qui permet de **remplir un tableau** puis de **calculer et d'afficher les carrés des éléments** du tableau
- 2) Effectuez une rotation circulaire sur le tableau, le pas de rotation est saisi au clavier

Exercice 4:

Parmi tous les entiers supérieurs à 1, seuls 4 peuvent être représentés par la somme des cubes de leurs chiffres. A titre d'exemple, $153 = 1^3 + 5^3 + 3^3$ est un nombre cubique. Écrivez un bloc pl/sql permettant de déterminer les 3 autres.

Note : les 4 nombres sont compris entre 150 et 410.

Ingénierie des bases de données 19/01/2022 48



Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

La gestion des exceptions en PL/SQL

- Lorsqu'une instruction se passe mal en PL/SQL, une exception est levée. Il est possible de spécifier un comportement adapté dans ce cas, dans la section **EXCEPTION**.
- Cela permet à l'exécution de se poursuivre si l'erreur n'est pas assez importante pour mettre fin au programme.
- Si une erreur est rencontrée et traitée de cette manière, l'exception est traitée, et le programme dépasse le bloc actuel et le processus d'exécution continue.

Ingénierie des bases de données 19/01/2022 50

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Types des exceptions en PL/SQL

- Types d'exceptions :
 - Les exceptions internes** : sont générées par le moteur du système (division par zéro, connexion non établie, table inexistante, privilèges insuffisants, mémoire saturée, espace disque insuffisant, ...). Une erreur interne est produite quand un bloc PL/SQL viole une règle d'Oracle ou dépasse une limite dépendant du système d'exploitation. Chaque erreur ORACLE correspond un code SQL **SQLCODE** et un message **SQLERRM**.
 - Les exceptions externes** : sont générées par l'utilisateur (stock à zéro, ...).

Ingénierie des bases de données 19/01/2022 51

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Types des exceptions en PL/SQL

EXCEPTION	DESCRIPTION	TRAITEMENT
Erreur pré-définie du oracle	Une des 20 erreurs qui arrivent le plus fréquemment en langage P/SQL	Ne pas la déclarer et laisser oracle serveur l'émettre implicitement
Erreur non pré-définie du oracle	Toutes les autres erreur standard d'oracle	La déclarer à l'intérieur de la section declare et laisser oracle l'émettre implicitement
Erreur définie par l'utilisateur	Une condition que le programmeur définit comme anomalie	La déclarer à l'intérieur de la section declare et son émission est explicite

Exceptions internes

Exception externe

Ingénierie des bases de données 19/01/2022 52

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Syntaxe: Interception des exceptions

```

EXCEPTION
WHEN exception 1 [OR exception 11 ...] THEN
statement 1;
statement 2;
...
[WHEN exception 2 [OR exception 22 ...] THEN
statement 3;
statement 4;
...]
[WHEN OTHERS THEN
statement 5;
...]
```

Ingénierie des bases de données 19/01/2022 53

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Règles d'interception des exceptions

- Plusieurs exceptions sont permises (définie et prédéfinie)
- Une seule exception est exécutée avant de sortir d'un bloc
- WHEN OTHERS est la dernière clause :
 - Intercepte toutes les exceptions non gérées dans la même section d'exception,
 - Utilisez le gestionnaire d'erreurs OTHERS et placez le en dernier lieu après tous les autres gestionnaire d'erreurs, sinon il interceptera toutes les exceptions mêmes celle qui sont prédéfinie.

Ingénierie des bases de données 19/01/2022 54

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Oracle prédéfinies:

Nom d'exception	Erreur ORACLE	SQLCODE
CURSOR_ALREADY_OPEN	ORA-06511	-6511
DUP_VAL_ON_INDEX	ORA-00001	-1
INVALID_CURSOR	ORA-01001	-1001
INVALID_NUMBER	ORA-01722	-1722
LOGIN_DENIED	ORA-01017	-1017
NO_DATA_FOUND	ORA-01403	+100
NOT_LOGGED_ON	ORA-01012	-1012
PROGRAM_ERROR	ORA-06501	-6501
PORTYPE_MISMATCH	ORA-06504	-6504
STORAGE_ERROR	ORA-06500	-6500
TIMEOUT_ON_RESOURCE	ORA-00051	-51
TOO_MANY_ROWS	ORA-01422	-1422
VALUE_ERROR	ORA-06502	-6502
ZERO_DIVIDE	ORA-01476	-1476

Ingenierie des bases de données 19/01/2022 55

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exemple

```

DECLARE
  v_sal emp.sal%type;
BEGIN
  SELECT sal INTO v_sal from emp;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE ('No Data Found');
  WHEN TOO_MANY_ROWS THEN
    DBMS_OUTPUT.PUT_LINE ('Invalid Data');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE ('Other errors');
  ROLLBACK ;
END ;

```

Ingenierie des bases de données 19/01/2022 56

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Oracle non-prédéfinies:

Declarative Section

Declare **Associate**

Nom de l'exception Association de l'exception
PRAGMA EXCEPTION_INIT

Exception Processing Section

Process

Traitement exécuté par l'exception

Ingenierie des bases de données 19/01/2022 57

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Oracle non-prédéfinies:

- On peut intercepter une erreur oracle non pré-définie en la déclarant au préalable.
- L'exception déclarée et implicitement déclenchée

- Déclarer le nom de l'exception oracle non-prédéfinie
Syntaxe
`nom _exception EXCEPTION;`
- Associer l'exception déclarée au code standard de l'erreur oracle
Syntaxe
`PRAGMA EXCEPTION_INIT (nom _exception , numero_erreur)`
- Traiter l'exception ainsi déclarée dans la section EXCEPTION.

Ingenierie des bases de données 19/01/2022 58

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Oracle non-prédéfinies:

- Exemple
Capturer l'erreur du Serveur Oracle numéro -2292 correspondant à la violation d'une contrainte d'intégrité.

```

DECLARE
  e_emps EXCEPTION;
  PRAGMA EXCEPTION_INIT (e_emps , -2292);
  v_deptno dept.deptno%type:=&p_deptno
BEGIN
  Delete from dept Where deptno=v_deptno;
EXCEPTION
  WHEN e_emps THEN dbms_output.put_line('Suppression impossible du dept
: ' || to_char(v_deptno) || ' employés existant ');
END;

```

Ingenierie des bases de données 19/01/2022 59

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exercices

- Soit la table suivante :
METEO(NOM_VILLE, Température, Humidité)
Écrire un bloc PL/SQL qui prends en entrée le nom d'une ville et retourne la température et l'humidité de cette ville. Gérer aussi par une exception le cas où la ville n'existe pas.

Ingenierie des bases de données 19/01/2022 60

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Utilisateurs

- PL/SQL permet à l'utilisateur de définir ses propres exceptions.
- La gestion des anomalies utilisateur peut se faire dans un bloc PL/SQL en respectant les étapes suivantes :
 - Sont définies dans la section **DECLARE**,
 - Sont déclenchées explicitement dans la section **BEGIN** par l'instruction **RAISE**,
 - Dans la section **EXCEPTION**, référencer le nom défini dans la section **DECLARE**.

Ingénierie des bases de données 19/01/2022 61

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Utilisateurs

Declarative Section
Declare
Nommer l'exception

Execution Section
ON
Activer
Déclencher l'exception explicitement en utilisant la clause **RAISE**

Exception Processing Section
Process
Définir le traitement de l'exception

Ingénierie des bases de données 19/01/2022 62

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Utilisateurs

- Nommer l'erreur (type exception) dans la partie **Declare** du bloc
Syntaxe
DECLARE
nom_ano EXCEPTION;
- Déterminer l'erreur et passer la main au traitement approprié par la commande **Raise**
Syntaxe
BEGIN
If (condition_anomalie) then raise nom_ano ;
- Effectuer le traitement défini dans la partie **EXCEPTION** du bloc.
Syntaxe
EXCEPTION
WHEN (nom_ano) then (traitement);

Ingénierie des bases de données 19/01/2022 63

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exceptions Utilisateurs

- Exemple

```

DECLARE
  x NUMBER;
  very_small_x EXCEPTION;
BEGIN
  .....
  IF x < 5 THEN RAISE very_small_x;
  END IF;
  .....
EXCEPTION
  WHEN very_small_x THEN dbms_output.put_line('La valeur de x est trop petite');
END;
```

Ingénierie des bases de données 19/01/2022 64

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exercices

Ecrivez des blocs PL/SQL qui permettent de:

- Enregistrer une livraison pour un produit donné :
 - saisie du numéro de produit et de la quantité livrée,
 - accès au stock du produit : exception si 'référence inconnue',
 - calcul du nouveau stock et mise à jour du stock.
- Déterminer le nombre de produit pour un fournisseur donné :
 - demande un numéro de fournisseur,
 - accès aux N° des produits de ce fournisseur,
 - retourne le message 'le fournisseur n'a pas de produit' ou 'le fournisseur a plusieurs produits', ou le N° du produit s'il n'en a qu'un.
- Accéder au stock d'un produit de Numéro donné avec Erreur Oracle quand le produit n'existe pas et Erreur utilisateur quand le stock est nul.
- Remplacer l'exception Oracle prédéfinie « NO_DATA_FOUND » par une exception utilisateur.

Ingénierie des bases de données 19/01/2022 65

CH 6
Les Déclencheurs

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Qu'est ce qu'un déclencheur (Trigger)?

Insert/update/delete

Avant insert/update/delete
Vérifier la date du traitement (le jour de la semaine/ l'heure...)

TABLE EMPLOYÉ

Après insert/update/delete
Enregistrer l'historique des transactions dans une autre table (audit)

Ingénierie des bases de données 19/01/2022 67

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Qu'est ce qu'un déclencheur (Trigger)?

- Les triggers sont des simples procédures, stockées dans la base de données, qui s'exécutent automatiquement lorsqu'une instruction INSERT, UPDATE ou DELETE (IUD) porte sur la table,

```
CREATE [OR REPLACE] TRIGGER <trigger_name>
[BEFORE | AFTER] [INSERT [OR] DELETE [OR] UPDATE]
ON <table_name>
[FOR EACH ROW][WHEN <condition>]
DECLARE
BEGIN
EXCEPTION
END ;
/
```

Ingénierie des bases de données 19/01/2022 68

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Types de Triggers

- 1) Triggers de table (STATEMENT):
 - Sont déclenchés une seule fois.
- 2) Triggers de ligne (ROW):
 - Se déclenchent individuellement pour chaque ligne de la table affectée par le trigger,

- Si l'option **FOR EACH ROW** est spécifiée, c'est un trigger ligne, sinon c'est un trigger de table.

Ingénierie des bases de données 19/01/2022 69

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Composants

- L'option **AFTER/BEFORE** précise le moment quand ORACLE déclenche le trigger,
- Le type d'instruction SQL qui déclenche le trigger:
 - DELETE, INSERT, UPDATE
 - On peut en avoir une, deux ou les trois
- Pour **UPDATE**, on peut spécifier une liste de colonnes. Dans ce cas, le trigger ne se déclenchera que si l'instruction UPDATE porte sur l'une au moins des colonnes précisées dans la liste.
- S'il n'y a pas de liste, le trigger est déclenché pour toute instruction UPDATE portant sur la table.

Ingénierie des bases de données 19/01/2022 70

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Composants

- Pour les triggers lignes, on peut introduire une restriction sur les lignes à l'aide d'une expression logique SQL : c'est la clause **WHEN** :
 - Cette expression est évaluée pour chaque ligne affectée par le trigger.
 - Le trigger n'est déclenché sur une ligne que si l'expression **WHEN** est vérifiée pour cette ligne.

Ingénierie des bases de données 19/01/2022 71

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Le corps du Trigger

- Le corps du trigger est un bloc PL/SQL :
 - Il peut contenir du SQL et du PL/SQL,
 - Il est exécuté si l'instruction de déclenchement se produit et si la clause de restriction **WHEN**, le cas échéant, est évaluée à vrai.

Ingénierie des bases de données 19/01/2022 72

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exemple

```
CREATE OR REPLACE TRIGGER StartInvoice
AFTER INSERT ON Invoice
FOR EACH ROW
DECLARE
    v_nb_insert NUMBER := 1;
BEGIN
    SELECT Nb_Insert INTO v_nb_insert
    FROM Statistics
    WHERE Table_name = 'Invoice';
    UPDATE Statistics
    SET Nb_Insert = v_nb_insert + 1
    WHERE Table_name = 'Invoice';
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        INSERT INTO Statistics VALUES ('Invoice',1);
END;
```

Ingénierie des bases de données 19/01/2022 75

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exercice

- Soit la table log suivante:
log(nom_table, date, opération)
 - Écrire un trigger qui permet de sauvegarder dans la table log toutes les mises à jours et les insertions faites sur la table employé.
- Écrire un trigger qui permet de vérifier l'heure et le jour d'exécution d'un traitement (insert/update ou delete) sur la table Département. Si le traitement est exécuté hors les heures de travail, une erreur est générée. Les heures de travail sont de 8h à 17h, du lundi au vendredi.

Ingénierie des bases de données 19/01/2022 74

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

OLD/NEW

- Lors de la création de **triggers lignes**, il est possible d'avoir accès à la valeur ancienne et la valeur nouvelle grâce aux mots clés **OLD** et **NEW**.
- Il n'est pas possible d'avoir accès à ces valeurs dans les triggers de table.
- Si l'instruction de déclenchement du trigger est INSERT, seule la nouvelle valeur a un sens.
- Si l'instruction de déclenchement du trigger est DELETE, seule l'ancienne valeur a un sens.

Ingénierie des bases de données 19/01/2022 76

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

OLD/NEW

- La nouvelle valeur est appelée **:new.colonne**
- L'ancienne valeur est appelée **:old.colonne**

➤ Exemple

```
CREATE OR REPLACE TRIGGER MAJ_Sal
BEFORE UPDATE ON Employe
FOR EACH ROW
BEGIN
    DBMS_OUTPUT.PUT_LINE('Avant la mise à jour ' || TO_CHAR(:OLD.sal) ||
    'vers ' || TO_CHAR(:NEW.sal));
END;
```

Ingénierie des bases de données 19/01/2022 74

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Exercice

- Écrire un trigger qui permet de contrôler les salaires : ils doivent être plus grand que le plus petit des salaires et plus petit que le plus grand des salaires

Ingénierie des bases de données 19/01/2022 77

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

```
CREATE OR REPLACE TRIGGER CTRLsalaire
BEFORE
INSERT OR UPDATE OF salaire
ON Employe
FOR EACH ROW
DECLARE
    minsal number;
    maxsal number;
BEGIN
    /* retrouver le salaire minimum et maximum dans la table employes */
    SELECT MIN(salaire), Max(salaire)
    INTO minsal, maxsal
    FROM Employe;

    /* s'il y a un problème, on provoque une erreur */
    IF (:new.salaire < minsal OR :new.salaire > maxsal) THEN
        raise_application_error (-20300, 'Salaire incorrect');
    END IF;
END;
```

Ingénierie des bases de données 19/01/2022 78

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

INSERTING/UPDATING/DELETING

- Quand un trigger comporte plusieurs instructions de déclenchement (par exemple INSERT OR DELETE OR UPDATE), on peut utiliser des prédicats conditionnels (INSERTING, DELETING et UPDATING) pour exécuter des blocs de code spécifiques pour chaque instruction de déclenchement.

Ingénierie des bases de données 19/01/2022 79

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

INSERTING/UPDATING/DELETING

- Syntaxe:

```
CREATE OR REPLACE TRIGGER nom_trigger
AFTER INSERT OR DELETE OR UPDATE ON nom_table
FOR EACH ROW
BEGIN
    IF inserting THEN
        ...
    END IF;
    IF deleting THEN
        ...
    END IF;
    IF updating THEN
        ...
    END IF;
END;
```

Ingénierie des bases de données 19/01/2022 80

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

INSERTING/UPDATING/DELETING

- Exemple:
Ecrire un trigger qui met à jour la table Class suite à une insertion ou suppression d'un étudiant dans la table Student:

```
Class(Id_Class, Nb_Stu)
Student(Id_Stu, Id_Class)
```

Ingénierie des bases de données 19/01/2022 81

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

INSERTING/UPDATING/DELETING

- Exemple:

```
CREATE OR REPLACE TRIGGER UpdateNbStudents
AFTER INSERT OR DELETE ON Student
FOR EACH ROW
BEGIN
    IF inserting THEN
        UPDATE Class
        SET Nb_Stu := Nb_Stu + 1
        WHERE Id_Class = :New.Id_Class;
    END IF;
    IF deleting THEN
        UPDATE Class
        SET Nb_Stu := Nb_Stu - 1
        WHERE Id_Class = :Old.Id_Class;
    END IF;
END;
```

Ingénierie des bases de données 19/01/2022 82

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Gestion des déclencheurs

- Activer ou désactiver un déclencheur :
`ALTER TRIGGER <nom_déclencheur> [ENABLE | DISABLE];`
- Supprimer un déclencheur :
`DROP TRIGGER <nom_déclencheur>;`
- Identifier un déclencheur dans une base de données :
`SELECT <nom_déclencheur> FROM déclencheurs_utilisateurs;`

Ingénierie des bases de données 19/01/2022 83



CH 7
Les fonctions et les procédures

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Définition

- Un sous-programme est un ensemble d'instructions PL/SQL qui porte un nom.
- Il existe deux types de sous-programmes :
 - Les procédures
 - Les fonctions
- Une procédure est un sous-programme qui exécute un code PL/SQL et ne retourne pas de résultat.
- Une fonction est un sous-programme qui exécute un code PL/SQL et renvoie un résultat.

Ingénierie des bases de données 19/01/2022 85

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les procédures

```

DECLARE
...
PROCEDURE <nom_proc> [(P1,...,Pn)] IS
  [Déclaration des variables locales]
BEGIN
...
EXCEPTION
...
END;
BEGIN
...
EXEC <nom_proc>[(A1,...,An)];
EXCEPTION
...
END;
  
```

Syntax of P1...Pn:
 <nom_arg>[IN | OUT | IN OUT] <Type>
 Avec:
 IN: input (par défaut)
 OUT: output
 IN OUT: Input/Output

Ingénierie des bases de données 19/01/2022 86

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les procédures: Exemple

- Soit la table Employé suivante:
 Employé(#id_Emp, nom, prénom, poste, salaire)
- Ecrire une procédure qui permet de mettre à jour le salaire d'un employé donné. Le nouveau salaire et l'id de l'employé sont passés en paramètres à la procédure.

Ingénierie des bases de données 19/01/2022 87

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les fonctions

```

DECLARE
...
FUNCTION <nom_fonc> [(P1,...,Pn)] RETURN Type IS
  [Local declarations]
BEGIN
...
RETURN value;
EXCEPTION
...
END;
BEGIN
...
var := <nom_fonc>[(A1,...,An)];
EXCEPTION
...
END;
  
```

Ingénierie des bases de données 19/01/2022 88

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Les fonctions: Exemple

We are going to create a function named `try_parse` that parses a string and returns a number if the input string is a number or `NULL` if it cannot be converted to a number.

```

1 CREATE OR REPLACE FUNCTION try_parse(
2   lv_number IN VARCHAR2)
3   RETURN NUMBER IS
4 BEGIN
5   RETURN to_number(lv_number);
6 EXCEPTION
7   WHEN others THEN
8     RETURN NULL;
9 END;
  
```

Let's create an anonymous block to use the `try_parse()` function.

```

1 SET SERVEROUTPUT ON SIZE 1000000;
2 DECLARE
3   n_x number;
4   n_y number;
5   n_z number;
6 BEGIN
7   n_x := try_parse('574');
8   n_y := try_parse('12.25');
9   n_z := try_parse('abc');
10
11 DBMS_OUTPUT.PUT_LINE(n_x);
12 DBMS_OUTPUT.PUT_LINE(n_y);
13 DBMS_OUTPUT.PUT_LINE(n_z);
14 END;
15 /
  
```

Ingénierie des bases de données 19/01/2022 89

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Procédures stockées et fonctions stockées

- Sont des blocs PL / SQL qui ont des noms.
- Sont utilisés pour stocker un bloc PL / SQL compilé dans la base de données.
- Peuvent être réutilisés sans être recompilés.
- Peuvent être appelés à partir de n'importe quel bloc PL/SQL.

Ingénierie des bases de données 19/01/2022 90

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Procédure stockée: Syntaxe

```
CREATE OR REPLACE PROCEDURE <nom_proc> [(P1,...,Pn)]
IS
[Déclaration des variables locales]
BEGIN
...
EXCEPTION
...
END;
```

Ingénierie des bases de données 19/01/2022 91

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Procédure stockée: Exemple

```
CREATE OR REPLACE PROCEDURE AddProd (PrefPro Prod.RefPro%TYPE,..., PPriUni
Prod.PriUni%TYPE, PErr OUT Number)
IS
BEGIN
INSERT INTO Prod VALUES(PrefPro,...,PPriUni) ;
COMMIT ;
PErr := 0 ;
EXCEPTION
WHEN OTHERS THEN
PErr := 1;
END;
```

Ingénierie des bases de données 19/01/2022 92

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Procédure stockée: Exemple

```
DECLARE
VErr NUMBER;
BEGIN
AddProd(123, ..., 500, VErr);
IF VErr = 0 THEN
DBMS_OUTPUT.PUT_LINE ('Operation Performed ');
ELSE DBMS_OUTPUT.PUT_LINE ('error');
END IF ;
END ;
```

Ingénierie des bases de données 19/01/2022 93

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Fonctions stockées: Syntaxe

```
CREATE [OR REPLACE] FUNCTION <nom_fonc> [(P1,...,Pn)]
RETURN Type IS
[Déclaration des variables locales]
BEGIN
...
RETURN(Valeur)
EXCEPTION
...
END;
```

Ingénierie des bases de données 19/01/2022 94

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Fonction stockée: Exemple

```
CREATE OR REPLACE FUNCTION NbEmp (PNumDep Emp.Dept_Id% Type, PErr OUT Number)
RETURN Number IS
VNB Number(4);
BEGIN
SELECT Count (*) INTO VNB FROM Emp
WHERE Dept_Id = PNumDep;
PErr := 0;
RETURN VNB;
EXCEPTION
WHEN OTHERS THEN
PErr := 1;
RETURN Null;
END;
```


Ingénierie des bases de données 19/01/2022 95

Introduction Bloc PL/SQL Variables Structures conditionnelles et itératives Gestion des erreurs Déclencheurs Fonctions et procédures

Fonction stockée: Exemple

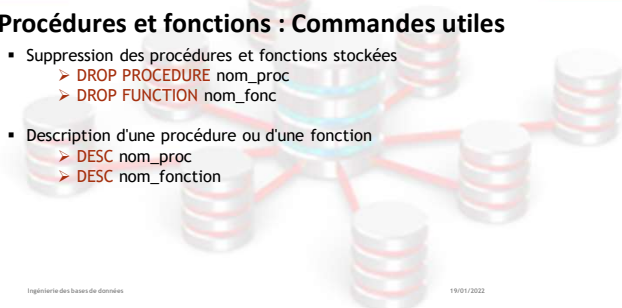
```
DECLARE
VErr NUMBER;
VNB NUMBER(4);
BEGIN
VNB := NbEmp(10, VErr);
IF VErr = 0 THEN
DBMS_OUTPUT.PUT_LINE ('The number of employees is
'||VNB);
ELSE DBMS_OUTPUT.PUT_LINE ('error');
END IF ;
END ;
```

Ingénierie des bases de données 19/01/2022 96




Procédures et fonctions : Commandes utiles

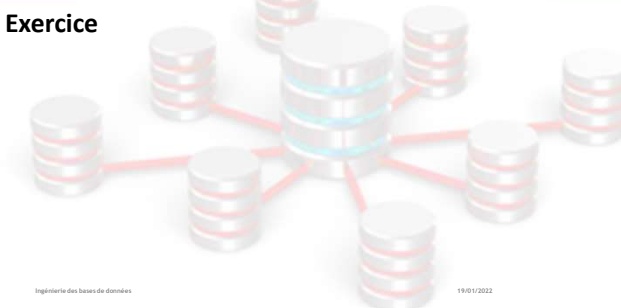
- Suppression des procédures et fonctions stockées
 - `DROP PROCEDURE nom_proc`
 - `DROP FUNCTION nom_fonc`
- Description d'une procédure ou d'une fonction
 - `DESC nom_proc`
 - `DESC nom_fonction`



Ingenierie des bases de données 19/01/2022 97



Exercice



Ingenierie des bases de données 19/01/2022 98