



ORAE505

Conception et administration de B.D.

5. Le langage PL/SQL

a. Les bases

Carina Roels

5.a.1 Généralités

SQL

Langage NON procédural

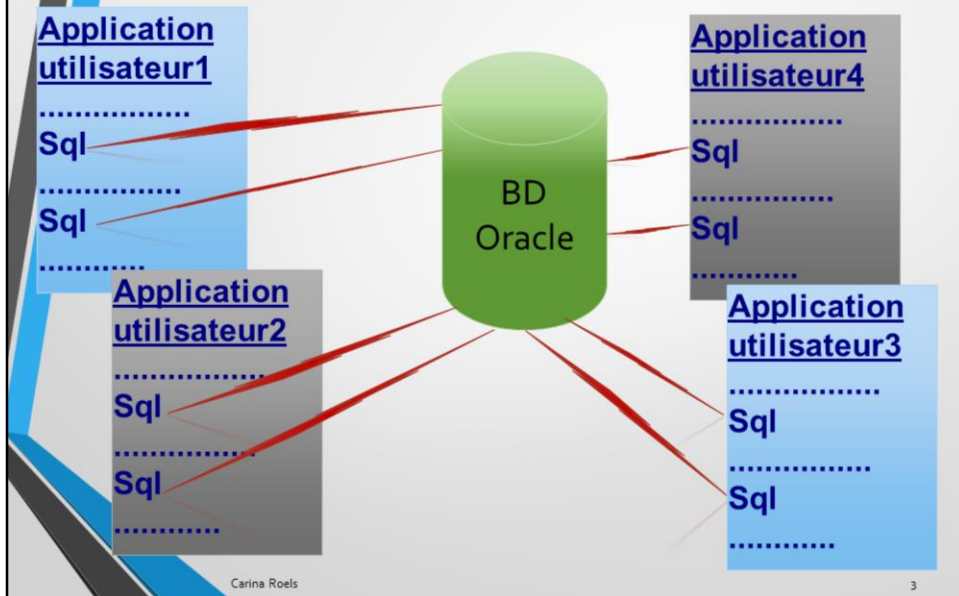
SELECT
INSERT
UPDATE
DELETE
COMMIT
ROLLBACK
...

PL-SQL

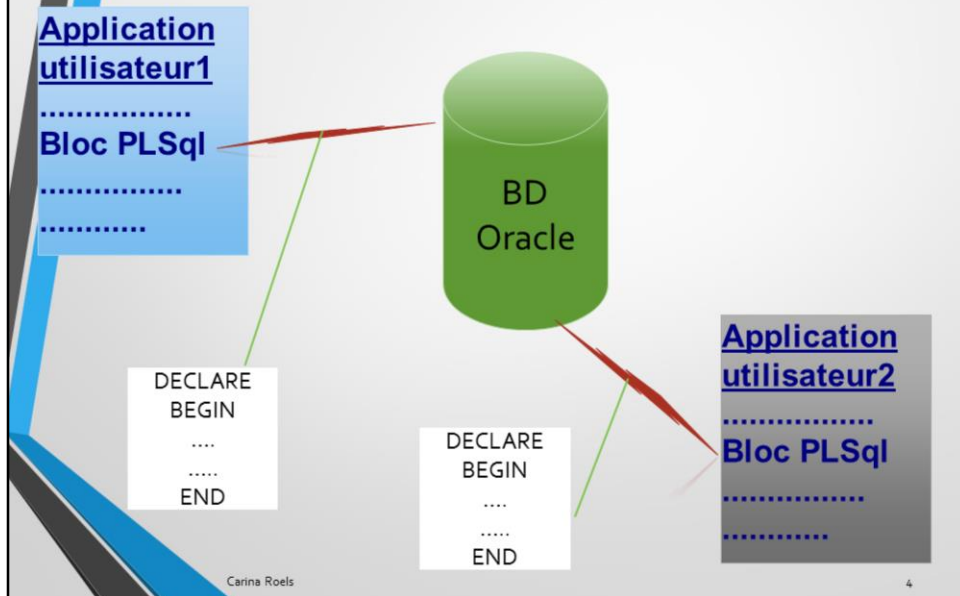
Langage procédural

Déclaration de variables
Traitements conditionnels
Traitements répétitifs
Gestion de curseurs
Gestion d'exceptions
Intégration d'instructions
SQL

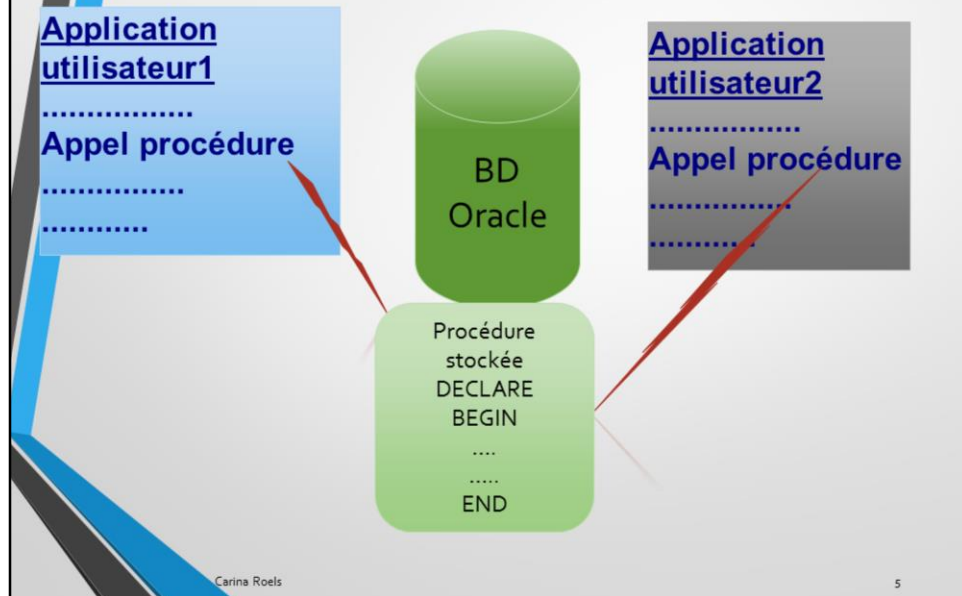
5.a.2 Fonctionnement - SQL



5.a.2 Fonctionnement - PL-SQL



5.a.2 Fonctionnement - objets stockés



5.a.3 La structure d'un bloc PL/SQL

Un bloc PL/SQL est interprété en une seule fois.
Il est composé de 3 sections.

DECLARE

...

...

Déclaration de variables, de
constantes, de curseurs ...

BEGIN

...

...

...

Corps du bloc PL/SQL, contenant un
mélange d'instructions PL/SQL et de
requêtes SQL.

EXCEPTION

...

END;

/

Traitement des exceptions et
d'erreurs.

Carina Roels

6

Les sections DECLARE et EXCEPTION sont facultatives.

Chaque instruction doit se terminer par un ;

Il est possible d'introduire des commentaires dans un bloc PL/SQL

-- sur une ligne

/* sur plusieurs
lignes */

Le bloc PL/SQL est exécuté par un / se trouvant à fin du code (en première position d'une ligne).

5.a.3 La structure d'un bloc PL/SQL (suite)

Il est possible d'imbriquer les blocs PL/SQL.
Cela sera particulièrement utile lors de la gestion des exceptions.

```
DECLARE
...
BEGIN
....
....
EXCEPTION
...
END;
```

```
DECLARE
...
BEGIN
...
....
...
END;
```

5.a.4 La déclaration de variables

Les variables locales au bloc PL/SQL

- variables de type ORACLE (char, varchar, number, date ...)
- variables de type BOOLEAN
- variables faisant référence au dictionnaire de données.

Les variables externes au bloc PL/SQL

les champs d'écran (SQL*Forms)

les variables en langage hôte (Pro*)

préfixées par :

les variables SQL*Plus

préfixées par &

5.a.4 La déclaration de variables

DECLARE

nom	vvarchar(20);
coddep	char(4);
salaire	number(8,2);
datemb	date;
reponse	boolean;

Référence au dictionnaire de données: nomemp prendra le.

nomemp	employe.nomemp%TYPE;
salemp	employe.salemp%TYPE;

Référence au dictionnaire de données: vdept sera une structure qui comportera des variables ayant le même nom et le même format que les colonnes de la table dept.

vdept	dept%ROWTYPE ;
-------	----------------

5.a.4 La déclaration de variables

Initialisation par affectation de valeur

DECLARE

nom	varchar(20)	:=	'Dupont';
Coddep	char(4)	:=	1234;
salaire	number(8,2)	:=	6500.99;
datemb	date	:=	'25/03/1998';
reponse	boolean	:=	TRUE;

Initialisation par select

```
select nomemp, preemp, salem  
into vnom, vprenom, vsal  
from employes  
where .....
```

Carina Roels

10

La clause **INTO** est obligatoire !

Le select doit ramener **1 seule** ligne de résultat

(si non, il faut utiliser un curseur, cf. chapitre suivant).

5.a.5 Les traitements conditionnels

```
If condition then  
  traitement  
End if ;
```

```
If condition then  
  traitement  
Else traitement2  
End if ;
```

```
If condition1 then  
  traitement1  
Elsif condition2 then  
  traitement2  
Elsif condition3 then  
  traitement3  
Else traitement4  
End if ;
```

Carina Roels

11

Les opérateurs utilisés dans les conditions d'un traitement conditionnel sont les mêmes que dans les requêtes SQL :

=	<	>	!=	>=	<=
IS NULL	IS NOT NULL	BETWEEN			
LIKE	AND		OR		
...					

5.a.6 Les traitements répétitifs

a) La boucle de base

```
BEGIN
...
LOOP
  EXIT WHEN
condition_de_sortie;
  Instructions PL/SQL et SQL
  .....
  .....
END LOOP;
...
END;
/
```

Carina Roels

12

Exemple : Afficher la table de multiplication du chiffre 5.

```
DECLARE
  I          number(2)    := 1;
  Chiffre    number(1)    := 5;
  Result     number(2);

BEGIN

  LOOP
    Result := Chiffre * I;
    INSERT INTO RESULTAT ( chaine )
    VALUES ( I || ' x ' || chiffre || ' = ' || result );

    I := I + 1;
    EXIT WHEN I > 10;

  END LOOP;
END;
/

select * from resultat;
```

5.a.6 Les traitements répétitifs

b) La boucle FOR

```
BEGIN
  FOR indice IN expression1 ..
    expression2
  LOOP
    Instructions PL/SQL et SQL
    .....
    .....
  END LOOP;

END;
/
```

Carina Roels

13

Exemple : Afficher la table de multiplication du chiffre 5.

```
DELARE
  Chiffre      number(1)  := 5 ;
  Result       number(2);
BEGIN
  FOR I IN 1 .. 10
  LOOP
    Result := Chiffre * I;

    INSERT INTO RESULTAT ( chaine )
    VALUES ( I || ' x ' || chiffre || ' = ' || result );

  END LOOP;
END;
/

select * from resultat ;
```

5.a.6 Les traitements répétitifs

c) La boucle WHILE

```
BEGIN
  WHILE condition
  LOOP
    Instructions PL/SQL et
    SQL
    .....
    .....
  END LOOP;
END;
/
```

Dans la condition, toutes les expressions SQL sont valables :

< > = != AND OR
LIKE ...

LIKE ...
< > = != AND OR

Carina Roels

14

Exemple : Afficher la table de multiplication du chiffre 5.

```
DECLARE
  I          number(2)   := 1;
  Chiffre    number(1)   := 5;
  Result     number(2);
BEGIN
  WHILE I <= 10
  LOOP
    Result := Chiffre * I;

    INSERT INTO RESULTAT ( chaine )
    VALUES ( I || ' x ' || chiffre || ' = ' || result );

    I := I + 1;
  END LOOP;
END;
/

select * from resultat ;
```