Définition du PL/SQL en Oracle

PL/SQL (Procedural Language extensions to SQL) est le langage de programmation procédural développé par Oracle pour son système de gestion de base de données relationnelle (SGBD).

Caractéristiques principales du PL/SQL :

1. **Langage procédural** : Contrairement au SQL standard qui est déclaratif, PL/SQL permet d'écrire des structures de programmation comme des boucles, des conditions et des procédures.
2. **Intégration avec Oracle** : Spécifique à Oracle, il est parfaitement intégré avec le moteur de base de données Oracle.
3. **Structures de blocs** : Le code PL/SQL est organisé en blocs (anonymes ou nommés) avec une structure :

```
DECLARE
  -- Déclaration des variables
BEGIN
  -- Corps exécutable
EXCEPTION
  -- Gestion des erreurs
END;
```

4. **Types d'objets** : Permet de créer des :
    - Procédures stockées
    - Fonctions
    - Packages (collections de procédures/fonctions)
    - Déclencheurs (triggers)
    - Types objets (programmation orientée objet)
5. **Performances** : Exécuté directement par le serveur Oracle, réduisant les allers-retours réseau par rapport à l'envoi de multiples commandes SQL.

PL/SQL combine la puissance du SQL pour manipuler les données avec les capacités d'un langage de programmation pour traiter ces données de manière complexe.

SQL*Plus et son environnement Oracle

Rappel sur SQL*Plus

SQL*Plus est l'outil en ligne de commande historique d'Oracle pour interagir avec la base de données. Bien que rudimentaire, il reste puissant pour l'administration et l'exécution de scripts.

Problèmes classiques

1. Problèmes de connexion (TNSNAMES.ORA)

- **Erreur "ORA-12154: TNS:could not resolve the connect identifier specified"**
  - Vérifier que le fichier tnsnames.ora existe dans $ORACLE_HOME/network/admin
  - Vérifier la syntaxe des entrées dans tnsnames.ora
  - Vérifier la variable d'environnement TNS_ADMIN si utilisée

2. Base arrêtée

- **Erreur "ORA-01034: ORACLE not available"**
  - Vérifier l'état de la base: sqlplus / as sysdba puis STARTUP
  - Consulter les logs d'alertes

3. Problèmes de listener

- **Erreur "ORA-12541: TNS:no listener"**
  - Vérifier que le listener est démarré: lsnrctl status
  - Démarrer si nécessaire: lsnrctl start
  - Vérifier le fichier listener.ora

Utilisation efficace du buffer des commandes

- L ou LIST: Affiche le contenu du buffer
- /: Exécute la commande du buffer
- A[PPEND] texte: Ajoute du texte à la ligne courante
- C[HANGE] /ancien/nouveau/: Remplace du texte
- DEL: Supprime la ligne courante
- SAVE nom_fichier: Sauvegarde le buffer dans un fichier
- GET nom_fichier: Charge un fichier dans le buffer
- EDIT: Ouvre l'éditeur par défaut

Mise en forme des résultats

```
// Largeur des colonnes
COLUMN nom_colonne FORMAT A20
COLUMN salaire FORMAT 9999.99

// Titres
TTITLE 'Rapport des employés'
BTITLE 'Fin du rapport'

// Pagination
SET PAGESIZE 50
```

```
SET LINESIZE 120

// Suppression des espaces superflus
SET TRIMSPOOL ON
```

Stocker les résultats dans un fichier (SPOOL)

```
SPOOL rapport_employes.lst
// Vos commandes SQL ici
SPOOL OFF
```

Scripts interactifs

```
// Demande une valeur à l'utilisateur
ACCEPT dept PROMPT 'Entrez le numéro de département: '

// Utilisation dans une requête
SELECT * FROM emp WHERE deptno = &dept;

// Définition de variable
DEFINE mon_var = 10
SELECT * FROM emp WHERE sal > &&mon_var;
```

Positionnement des outils Oracle

1. **SQL*Plus**:
    - Outil en ligne de commande basique
    - Idéal pour l'administration, l'exécution de scripts batch
    - Léger, disponible partout
2. **SQL Developer**:
    - IDE graphique gratuit d'Oracle
    - Plus convivial avec éditeur, debugger PL/SQL
    - Fonctionnalités avancées (modélisation, rapports)
3. **SQL**:
    - Langage déclaratif pour interroger et manipuler les données
    - Standard ANSI (avec extensions Oracle)
4. **PL/SQL**:
    - Langage procédural Oracle
    - Pour la logique métier dans la base (procédures, triggers)
    - Exécuté côté serveur

Exécuter un fichier SQL externe dans SQL*Plus

Il existe plusieurs méthodes pour exécuter un script SQL externe dans SQL*Plus :

Méthode 1 : À partir de la ligne de commande

```
sqlplus utilisateur/motdepasse@base_de_donnees @chemin/vers/le/fichier.sql
```

Ou pour exécuter en tant que SYSDBA :

```
sqlplus / as sysdba @chemin/vers/le/fichier.sql
```

Méthode 2 : Depuis l'invite SQL*Plus

```
SQL> @chemin/vers/le/fichier.sql
```

Si le fichier est dans le répertoire courant, vous pouvez simplement faire :

```
SQL> @fichier.sql
```

Méthode 3 : Avec la commande START

```
SQL> START chemin/vers/le/fichier.sql
```

Méthode 4 : En utilisant SPOOL pour capturer la sortie

```
SQL> SPOOL resultats.log
SQL> @fichier.sql
SQL> SPOOL OFF
```

Méthode 5 : Pour les scripts avec paramètres

```
SQL> @fichier.sql param1 param2 param3
```

Dans le script, les paramètres sont accessibles via &1, &2, etc. :

```
//Contenu de fichier.sql
SELECT * FROM employes WHERE departement_id = &1;
```

1. Utilisez des chemins absolus pour éviter les problèmes de répertoire courant
2. Pour les chemins Windows, utilisez des doubles backslashes ou des slashs :
   ```
   @C:\\chemin\\vers\\fichier.sql
   -- ou
   @C:/chemin/vers/fichier.sql
   ```
3. Vérifiez les droits d'accès au fichier

4. Utilisez SET ECHO ON pour voir les commandes exécutées :

```
SET ECHO ON
@fichier.sql
```

Astuce supplémentaire

Pour connaître le répertoire courant dans SQL*Plus :

```
SQL> HOST pwd  (sur Unix/Linux)
SQL> HOST cd   (sur Windows)
```

PL/SQL : Principes Fondamentaux et Structure

1. Que signifie "PL" dans PL/SQL ?

PL signifie **Procedural Language**, une extension procédurale du langage **SQL**.

- **SQL** est déclaratif (on dit "quoi faire").
- **PL/SQL** est procédural (on dit "comment le faire").

2. Structure des blocs PL/SQL

Un bloc PL/SQL a la structure suivante :

```
[DECLARE
   // Déclarations de variables, constantes, curseurs, exceptions...
BEGIN
   // Instructions exécutables (SQL, boucles, conditions...)

[EXCEPTION    // Gestion des erreurs]

END;
```

- **DECLARE** (optionnel) : Définit les variables et objets.
- **BEGIN-END** (obligatoire) : Contient la logique exécutable.
- **EXCEPTION** (optionnel) : Gère les erreurs.

Exemple :

```
DECLARE
   v_nom VARCHAR2(50) := 'Samba';
   v_salaire NUMBER := 2500;

BEGIN

   DBMS_OUTPUT.PUT_LINE('Nom : ' || v_nom);
   DBMS_OUTPUT.PUT_LINE('Salaire : ' || v_salaire);

END;
```

3. Comment Oracle interprète un programme PL/SQL ?

1. **Parsing** : Vérification de la syntaxe.
2. **Compilation** : Conversion en code interprétable (p-code).
3. **Exécution** :
    - Le moteur PL/SQL exécute le code.
    - Les requêtes SQL sont envoyées au moteur SQL.
    - Les résultats sont renvoyés à PL/SQL.

4. Types de données et conversion

Types de base :

- **NUMBER** : Entiers/décimaux (INT, FLOAT).
- **VARCHAR2** : Chaînes de caractères (taille variable).
- **DATE** : Dates et heures.
- **BOOLEAN** : TRUE/FALSE/NULL (uniquement en PL/SQL).

Conversion de types :

- TO_CHAR() : Convertit en chaîne.
- TO_NUMBER() : Convertit en nombre.
- TO_DATE() : Convertit en date.

Exemple :

```
DECLARE

   v_date_str VARCHAR2(20) := '2023-10-25';
   v_date DATE;

BEGIN

   v_date := TO_DATE(v_date_str, 'YYYY-MM-DD');

END;
```

5. Cas particulier des dates

Oracle stocke les dates avec l'heure.

- **TO_DATE()** : Convertit une chaîne en date.
- **TO_CHAR()** : Formate une date en chaîne.
- **NLS_DATE_FORMAT** : Définit le format par défaut.

Exemple :

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD/MM/YYYY'; // Format français
SELECT TO_CHAR(SYSDATE, 'Day, DD Month YYYY') FROM DUAL; // "Mercredi, 25 Octobre 2023"
```

6. Liens entre requêtes SQL et variables PL/SQL

Les variables PL/SQL peuvent :

- **Stocker des résultats SQL** (SELECT INTO).
- **Être utilisées dans des requêtes** (WHERE colonne = v_variable).

Exemple : Ecrire un programme PL/SQL qui affiche le nom et le salaire de l'employé numéro 100. Employes (id, nom, salaire)

```
DECLARE

   v_emp_name VARCHAR2(100);
   v_emp_salary NUMBER;

BEGIN

   SELECT nom, salaire INTO v_emp_name, v_emp_salary
   FROM employes
   WHERE id = 100;

   DBMS_OUTPUT.PUT_LINE(v_emp_name || ' gagne ' || v_emp_salary);

END;
```

7. Instructions arithmétiques

Opérations standard : +, -, *, /, ** (puissance), MOD().

Exemple :

```
DECLARE

   v_a NUMBER := 10;
   v_b NUMBER := 3;
   v_result NUMBER;

BEGIN

   v_result := v_a * v_b + MOD(v_a, v_b);  -- 10 * 3 + (10 % 3) = 30 + 1 = 31
```

```
END;
```

8. Instructions conditionnelles (IF, ELSE, ELSIF)

Structure :

```
IF condition1 THEN
   -- Code
ELSIF condition2 THEN
   -- Code
ELSE
   -- Code par défaut
END IF;
```

Exemple :

```
DECLARE

   v_note NUMBER := 15;

BEGIN

   IF v_note >= 16 THEN
      DBMS_OUTPUT.PUT_LINE('Très bien');

   ELSIF v_note >= 12 THEN

      DBMS_OUTPUT.PUT_LINE('Bien');

   ELSE

      DBMS_OUTPUT.PUT_LINE('Insuffisant');

   END IF;
END;
```

9. Boucles (LOOP, FOR, WHILE)

Boucle simple (LOOP-EXIT WHEN) :

```
LOOP
   -- Code
   EXIT WHEN condition;
END LOOP;
```

Boucle FOR :

```
FOR i IN 1..10 LOOP
   DBMS_OUTPUT.PUT_LINE('Itération ' || i);
```

```
END LOOP;
```

Boucle WHILE :

```
WHILE condition LOOP
   -- Code
END LOOP;
```

10. Packages prédéfinis (DBMS_...)

- **DBMS_OUTPUT** : Affichage de résultats (PUT_LINE).
- **DBMS_SQL** : Exécution dynamique de SQL.
- **DBMS_JOB** / DBMS_SCHEDULER : Planification de tâches.
- **DBMS_LOB** : Manipulation de gros objets (BLOB, CLOB).
- **DBMS_RANDOM** : Génération de nombres aléatoires.

Exemple :

```
BEGIN

   DBMS_OUTPUT.PUT_LINE('On debute en, PL/SQL!');
   DBMS_RANDOM.SEED(123);
   DBMS_OUTPUT.PUT_LINE('Nombre aléatoire : ' || DBMS_RANDOM.VALUE(1, 100));

END;
```

Conclusion

- PL/SQL étend SQL avec des structures procédurales.
- Un bloc PL/SQL contient des déclarations, un corps exécutable et une gestion d'erreurs.
- Oracle compile et exécute le code via son moteur PL/SQL.
- Les packages (DBMS_...) fournissent des fonctions avancées.

Programme PL/SQL pour calculer la somme de deux valeurs saisies

1. Demande à l'utilisateur de saisir deux valeurs A et B
2. Vérifie que les valeurs ne sont pas nulles
3. Calcule et affiche la somme des deux valeurs

```
DECLARE

      v_a NUMBER;
   v_b NUMBER;
   v_somme NUMBER;
```

```
BEGIN
   v_a := &saisir_valeur_a;
   v_b := &saisir_valeur_b;

   IF v_a IS NULL OR v_b IS NULL THEN

      DBMS_OUTPUT.PUT_LINE('Erreur : Les valeurs A et B ne doivent pas être nulles');
   ELSE
      v_somme := v_a + v_b;

      -- Affichage du résultat
      DBMS_OUTPUT.PUT_LINE('Résultat :');
      DBMS_OUTPUT.PUT_LINE('A = ' || v_a);
      DBMS_OUTPUT.PUT_LINE('B = ' || v_b);
      DBMS_OUTPUT.PUT_LINE('A + B = ' || v_somme);

   END IF;

EXCEPTION
   WHEN VALUE_ERROR THEN
      DBMS_OUTPUT.PUT_LINE('Erreur : Veuillez saisir des nombres valides');
   WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('Erreur inattendue : ' || SQLERRM);
END;
/
```

Version améliorée avec boucle de resaisie

```
DECLARE
   v_a NUMBER;
   v_b NUMBER;
   v_saisie_valide BOOLEAN := FALSE;
BEGIN
   -- Boucle jusqu'à ce que la saisie soit valide
   WHILE NOT v_saisie_valide LOOP
      BEGIN
         -- Saisie des valeurs
         DBMS_OUTPUT.PUT_LINE('--- Calcul de A + B ---');
         v_a := &saisir_valeur_a;
         v_b := &saisir_valeur_b;

         -- Vérification des valeurs
         IF v_a IS NULL OR v_b IS NULL THEN
            DBMS_OUTPUT.PUT_LINE('Erreur : Les valeurs ne doivent pas être nulles');
         ELSE
            -- Affichage du résultat
            DBMS_OUTPUT.PUT_LINE('Résultat : ' || v_a || ' + ' || v_b || ' = ' || (v_a + v_b));
            v_saisie_valide := TRUE;
         END IF;
      EXCEPTION
```

```
        WHEN VALUE_ERROR THEN
          DBMS_OUTPUT.PUT_LINE('Erreur : Veuillez saisir des nombres valides');
        WHEN OTHERS THEN
          DBMS_OUTPUT.PUT_LINE('Erreur inattendue : ' || SQLERRM);
      END;
  END LOOP;
END;
/
```

Comment exécuter ce script

1. Dans SQL*Plus ou SQL Developer :
2. Activer l'affichage avec **SET SERVEROUTPUT ON**
3. Copier/coller le script
4. Lorsque le programme demande la saisie, entrer des valeurs numériques

Ce programme illustre :

- L'utilisation des variables de substitution (&)
- La gestion des entrées utilisateur
- Le contrôle des valeurs NULL
- Le traitement des exceptions
- Les opérations arithmétiques de base

Les Curseurs en PL/SQL : Guide Complet