Travail avec une BD sous oracle

Une fois que oracle est installé, on procède de la manière suivante pour travailler avec une base de données (ouvrir une instance Oracle, ...):

1. Se connecter sous votre compte LINUX :
login : xxxxxx
passwd : *****
user$

1. bis. Installer l'outil rlwrap (pour que les flèches d'historique fonctionnent sous sqlplus) :

user$ **sudo apt-get install rlwrap**
passwd : *****
user$

2. Se connecter sous le compte utilisateur oracle (du groupe dba) :

user$ **su – oracle**
passwd : ****** (mot de passe donné à la création du compte oracle
oracle$

3. Se connecter sous **sqlplu**s avec le compte : **sys as sysdba** :

oracle$ **rlwrap sqlplus sys as sysdba**

passwd : ***** (mot de passe du DBA sys sous oracle (demandé pendant l'installation)

SQL> --démarrer une instance oracle :

SQL> **startup**
Oracle instance started

Total System Global Area 1068937216 bytes
Fixed size
Variable size
Database buffers
Redo buffers
Database mounted
Database opened

SQL> -- Créer un utilisateur et donner des droits
SQL> -- en même temps connect, ressource, dba

SQL> **grant connect, ressource, dba to *mon_nom* identified by *mon_pass* ;**

SQL> -- Se déconnecter de sys et se connecter sous mon_nom :
SQL> **disconnect**

SQL> **connect *mon_nom* as sysdba**
passwd : *mon_pass*

SQL> -- On peut exécuter des scripts de création de tables et d'insertions initiales

SQL> **@demobld.sql**
-- est un exemple de script qui existe sous le répertoire courant -- (ou bien **start demobld.sql**)

SQL> -- Les tables sont crées et peuplées par des données : on peut commencer à manipuler : select, update, insert, delete, drop, alter, create, ..

SQL> -- Pour arrêter l'instance, il faut se déconnecter et se reconnecter sous sys as sysdba

SQL> **disconnect**

```
SQL> connect sys as sysdba
passwd : ***** (mot de passe du DBA sys sous oracle (demandé pendant l'intallation)

SQL> shutdown [normal | transactional | immediate | abort]
database closed
database dismounted
ORACLE instance shutdown <option
SQL>

Les options sont :

-- NORMAL : attend la déconnexions de toutes les sessions
-- déjà ouvertes et interdit de nouvelles connexions. C'est le
-- mode le plus lent.
-- Il est le mode par défaut, c-à-d c'est comme si shutdown
-- est lancé sans paramètres.

-- TRANSACTIONAL: attend que les transactions en cours
-- soient terminées. Mais pas de nouvelles transactions.

-- IMMEDIATE : n'attend pas que les transactions en cours
-- soient terminées. Les transactions non validées au moment
--du shutdown  sont annulées (abort). C'est l'option d'arrêt
--cohérent la plus rapide.

-- ABORT : c'est l'équivalent d'une coupure de courant :
--c'est le  mode d'arrêt le plus rapide, mais la base risque
--d'être  incohérente. En effet, les modifications non
--validées ne sont pas annulées et le cache des tampons de
--la base de données n'est pas écrit sur les fichiers de la
--base.


SQL> exit
oracle$ exit
user$
```

L'Outil SQLPLUS d'Oracle

ORACLE : Système de Gestion de Bases de Données relationnel

Dispose de plusieurs outils dont SQLPLUS.

SQLPLUS = outil d'Oracle qui permet :

- d'utiliser les commandes du langage standard SQL
- d'utiliser les commandes du langage procédural PL/SQL
- autres commandes

- ***SQL : permet de stocker, modifier, supprimer, retrouver des données stockées dans le SGBD.***
- ***PL/SQL : permet de regrouper des ordres SQL dans des blocs*** (procédures)

L'outil SQLPLUS permet donc :

- d'utiliser des ordres SQL

- d'utiliser des blocs PL/SQL

- d'exécuter d'autres tâches spécifiques; par exemple :

1. ***entrer, stocker, éditer, retrouver et exécuter des commandes***
   SQL et des blocs PL/SQL, stockées temporairement dans un
   Buffer SQL

2. ***formatter, effectuer des calculs, stocker et imprimer les résultats***

3. ***lister les descriptions des tables ou des colonnes***

4. ***accéder et échanger des données entre différentes B.D***

5. ***afficher des messages pour l'utilisateur (ou accepter des réponses)***

1. Eléments de SQLPLUS

DEFINITIONS :

- **commande** : instruction pour SQL*PLUS ou pour le SGBD

- **bloc** : ensemble de commandes SQL (ou PL/SQL) regroupées en une procédure logique

- **table** : unité de base de stockage dans le SGBD

- **interrogation** : commande SQL (un SELECT) pour retrouver des informations à partir d'une ou plusieurs tables.

EXEMPLES DE COMMANDES SQLPLUS:

- pour donner le label EMPLOYE à la colonne ENAME d'une table :

SQL> **COLUMN ENAME HEADING EMPLOYE**

- pour lister les description des colonnes d'une table (EMP) :

SQL> **DESCRIBE EMP**

TABLES UTILISEES comme EXEMPLES : EMP et DEPT.

LANCER SQL*PLUS : à partir de l'invite du système d'exploitation, taper :

$ SQLPLUS

==> ensuite donner le nom de "login" et le mot de passe
==> le message d'invite de SQLPLUS apparaît : SQL>

QUITTER SQL*PLUS : taper EXIT

DEMANDER de l'AIDE : HELP <nom-de-la-commande>

EXECUTER une COMMANDE :

taper le texte de la commande (sur une ou plusieurs lignes)

terminer par une point virgule puis <RC> (RETURN, ENTER)

FINIR une COMMANDE :
- avec le "**;**" à la fin de la commande puis <RC>
  ⇨ la commande est exécutée immédiatement

- avec un "**/**" tout seul sur une ligne, suivi de <RC>
  ⇨ exécution de la commande stockée dans le buffer

- avec **une ligne vide** suivi de <RC>
  ⇨ la commande est sauvegardée dans le buffer SQL, mais n'est pas exécutée.

Rmq : Le BUFFER SQL : SQL*PLUS stocke toujours la dernière commande SQL dans le buffer SQL. Le ";" ou le "/" n'y sont pas sauvegardés.

RUN : exécution de la commande contenue dans le buffer avec listage de la commande.

/ : exécution sans listage de la commande

Exemple de bloc PL/SQL :
A taper sous SQLPLUS ou fichier à charger ou à exécuter.

```
SQL> DECLARE
  2         x    NUMBER  := 100;
  3   BEGIN
  4       FOR  i  IN  1..10  LOOP
  5           IF  TRUNC(i / 2)  =  i / 2  THEN       -- i est pair
  6               INSERT INTO  temp VALUES (i, x, ' i est pair');
  7           ELSE
  8               INSERT  INTO temp VALUES (i, x, 'i est impair');
  9            END IF;
 10            x := x + 100;
 11        END LOOP;
 12   END;
 13   .
SQL>  /

PL/SQL  procedure successfully completed.

SQL>
```

EXEMPLE DE FORMATTAGE :

SQL> **COLUMN sal FORMAT $99,999 HEADING salaire**

SQL> **RUN** <-- réexécute la dernière commande d'interrogation

```
  1  SELECT ename, sal  FROM  emp

  ename              salaire
----------         ----------------
SMITH                    $800
ALLEN                  $1,600
WARD                   $1,250
etc ...
```

REMARQUE : Dans le buffer, ne sont sauvegardées que les ordres SQL
ou PL/SQL.

COMMANDES de l'EDITEUR de SQL*PLUS : servent à manipuler la commande SQL qui est dans le **buffer de SQLPLUS**

- **A** *texte* (APPEND texte) : ajoute *texte* à la fin de ligne

- **C** /*ancien*/*nouveau* (CHANGE ...) : change *ancien* par *nouveau* dans la ligne courante

- **C** /*texte* (CHANGE ..) : supprime *texte* de la ligne

- **CL** *BUFF* (CLEAR BUFFER) : vide le buffer

- **DEL** : supprime ligne courante

- **I** (INPUT) : ajoute une ou plusieurs lignes)

- **I** *texte* (INPUT texte) : ajoute du texte à la suite de la fin de ligne

- **L** (LIST) : affichage des lignes du buffer

- **L** *n* (LIST n ou n) : liste la nème ligne

- **L** * (LIST *) : liste la ligne courante

- **L** *LAST* (LIST LAST) : liste la dernière ligne

- **L** *m n* (LIST m n) : liste les lignes de n à m.

Sauvegarder, charger des FICHIERS de COMMANDES

SQL> **SAVE nom_fichier** : pour sauvegarder le contenu du buffer dans un fichier

SQL> **GET nom_fichier** : charger dans le buffer le contenu du Fichier

Note : par défaut, l'extension du fichier est «.SQL»

EXECUTER un fichier de commandes :

SQL> **START nom_fichie**r

Remarque : nom_fichier possède l'extension ".SQL" sauf indication contraire par l'utilisateur lors de la commande SAVE.

Exemples :
SQL> SAVE toto
===> sauvegarde la dernière commande qui est dans le buffer dans un fichier appelé toto.sql
Exécution :
SQL> START toto
ou
SQL> START toto.sql

Ou bien

SQL> SAVE titi.aa
==> Sauvegarde dans un fichier appelé titi.aa
Exécution :
SQL>START titi.aa

FORMATTAGE des COLONNES :

- **HEADING** : change le titre (le nom) de la colonne :

SQL> **COLUMN nom_colonne HEADING le_libelle**

exemple : SQL> COLUMN deptno HEADING departement;

on peut avoir le libellé sur 2 lignes :

SQL> **COLUMN ename HEADING 'Nom|Employe'**

==> résultat : Nom
Employe

- FORMATTAGE des valeurs de colonnes

SQL> **COLUMN nom_colonne FORMAT modele**

Exemples :

SQL> **COLUMN sal FORMAT $99,990**

==> afficher sal avec un signe $, une virgule, un 0 au lieu d'un blanc (là
où cela est nécessaire)

SQL> / <== le slash ré-exécute la dernière commande

Departement	Nom Employe	Salaire	Commission
30	ALLEN	$1,600	300
30	WARD	$1,250	500
30	MARTIN	$1,250	1400

etc ....

SQL> COLUMN ename FORMAT A4;

SQL> /

Departement	Nom Employe	Salaire	Commission
30	ALLEN	$1,600	300
30	WARD	$1,250	500
30	MART IN	$1,250	1400

etc ...

COLUMN nom_colonne : Informations sur une colonne

COLUMN : Informations sur toutes les colonnes

COLUMN nom_colonne CLEAR : Remettre les valeurs par défaut d'une colonne

CLEAR COLUMNS : REMETTRE les valeurs par DEFAUT (pour toutes les colonnes)

COLUMN nom_colonne OFF
COLUMN nom_colonne ON
=> SUPPRIMER / RESTORER un format pour une colonne

BREAK ON nom_colonne : supprime les valeurs dupliquées à l'affichage

Note : nom_colonne doit apparaître dans un ORDER BY (TRI), sinon résultat aléatoire.

```
SQL> BREAK ON deptno
SQL> SELECT deptno, ename, sal
       2  FROM  emp WHERE  sal < 2500
       3  ORDER BY deptno;
```

```
deptno     ename        sal
-------    --------     -------------
  10       CLARK         2450
           MILLER        1300
  20       SMITH          800
           ADAMS         1100
  30       ALLEN         1600
  etc ...
```

BREAK ON nom_col SKIP n

--> insère n lignes blanches au changement de valeur de nom_col.

Exemple. Sauter une ligne entre les départements :

```
SQL> BREAK ON  deptno  SKIP  1
SQL> /

    deptno     ename        sal
    -------   ----------    -------------
    10         CLARK         2450
               MILLER        1300

    20         SMITH          800
               ADAMS         1100

    30         ALLEN         1600
    etc ...
```

BREAK ON ROW SKIP n --> insère n lignes blanches après chaque
ligne (tuple) de la table

DEMANDER LA VALEUR D'UNE VARIABLE : &

Exemple : SQL> select * from emp where empno = **&num** ;
Enter value for num : **123**
La commande sera interprétée comme :
select * from emp where empno **= 123** ;

PARAMÉTRAGE DE LA COMMANDE START :

SQL>START fich1 CLERCK 7900

Dans le fichier fich1.sql, il y a la commande :

SELECT * FROM emp WHERE job = ‘&1’ AND sal = ‘&2’ ;

A l’exécution du fichier fich1, il y a substitution de &1 par CLERCK et de &2 par 7900.

INITIALISATION DE VARIABLES SYSTÈMES :

Ces variables contrôlent certaines conditions (affichages, formats, nombres de lignes, …)

SQL> SET AUTOCOMMIT ON -- dans un bloc PL/SQL, les modifications deviennent permanentes automatiquement (après chaque ordre SQL ou bloc PL/SQL)

SQL> SET AUTOCOMMIT OFF -- COMMIT explicite est nécessaire pour valider les modif.

SQL> SET PAGESIZE n -- nombre de lignes par écran

SQL> SET LINESIZE n -- nombre de caractères par ligne

SQL> SET PAUSE ON/OFF -- arrêt (ou non) du défilement après chaque page affichée.

SQL> SHOW variable_sql -- Afficher la valeur d’une variable :

Exemple :

SQL> SET PAUSE ON
SQL> show pause
linesize ON

SQL> **SHOW ALL** -- Pour connaître la valeur de toutes les variables SQL :

TRACE SUR UN FICHIER :

Pour garder sur un fichier la trace de toutes les commandes effectuées sous SQLPLUS et leurs résultats (erreurs comprises), on utilise la commande **SPOOL nom_fichier_trace**, au début et NE PAS OUBLIER de fermer le fichier trace, par la commande **SPOOL OFF**.

SQL> SPOOL FICHIER1.TRC
SQL> **select * from emp ;**
. . .
. . .

SQL> **select * from from ;**
Error : . . .
SQL> **select * from dept ;**
. . .
. . .
SQL> SPOOL OFF
SQL> select * from emp where empno=1786 ;
. . .
. . .
SQL>

Dans le fichier fichier1.trc, il y aura :

_____

select * from emp ;
. . .
. . .
select * from from ;
Error : ....
select * from dept ;
. . .
. . .
_____

Rmq. Si le fichier ne possède pas d'extension, l'extension par défaut du fichier trace est **.LST.**

Exemple :
SQL> spool nom_fich
……

SQL> spool off.

 fichier trace : **nom_fich.lst**

DIVERS :

Les commandes peuvent être saisies dès que le prompt SQL> est affiché.

SQLPlus est insensible à la casse (accepte indifféremment les majuscules et les minuscules). Attention aux chaînes de caractères : une chaîne de caractères ’texte’ es différente de ’Texte’ ou ’TEXTE’.

Interruption d’une commande SQL en cours :

deux possibilités : ctrl+C ou del.

Ecriture d’une commande SQL :

La saisie d’une instruction SQL peut être réalisée sur plusieurs lignes sans ponctuation particulière. L’instruction est stockée dans un buffer et les lignes
sont numérotées. Saisir une instruction sur plusieurs lignes permet une meilleure lisibilité et facilite sa modification ultérieure.

Rmq : On termine une instruction SQL par un point virgule pour qu’elle soit exécutée. Un simple retour à la ligne ne provoquera aucune exécution.

Demander à l’utilisateur d’entrer des valeurs :

Il est possible de demander à l’utilisateur d’entrer des valeurs, à l’aide du signe &.

Exemple :Pour une valeur numérique :

SELECT * FROM produit
WHERE **prix=&prix**;
Enter value for prix: **12**
old 1: select * from produit where prix=&prix
new 1: select * from produit where prix=12

Pour une chaîne de caractères, rajouter des guillemets :

SELECT * FROM produit
WHERE **marque='&marque'**;
Enter value for marque: **peugeot**
old 1: select * from produit where marque='&marque'
new 1: select * from produit where marque='peugeot'