Introduction à SQL sous ORACLE

Serge Tahé, université d'Angers, 1991

AVERTISSEMENT

Ce polycopié a été initialement écrit en 1991 et pratiquement pas remanié depuis. Certaines des informations qu'il contient sont désormais obsolètes. On trouvera un cours plus récent à l'Url [http://tahe.developpez.com/divers/sql-firebird/]. Il présente le langage SQL avec le SGBD libre Firebird.

ST, septembre 2001.

L'essentiel de l'ouvrage est tiré de la documentation officielle d'ORACLE. Cependant certains points sont inspirés de l'excellent ouvrage de Christian MAREE et Guy LEDANT : SQL Initiation, Programmation et Maîtrise paru chez EYROLLES.

SQL (*Structured Query Language*) est un langage standard de création, de maintenance et d'interrogation de bases de données relationnelles. Il est indépendant du SGBD utilisé. Si les exemples de ce document ont été écrits à l'aide du SGBD Oracle, ils peuvent cependant, pour la plupart, être reproduits avec tout SGBD relationnel. Sous Windows, on trouvera dans ce domaine outre Oracle des produits moins lourds tels Access, MySQL, SQL Server. Ils acceptent tous le langage SQL mais parfois avec des limites vis à vis de ce qu'accepte Oracle. Si de document est utilisé avec un autre produit qu'Oracle, on pourra ignorer toutes les informations spécifiques à ce SGBD, essentiellement celles concernant SQLPLUS l'outil d'Oracle pour interroger des bases avec SQL. Dans l'annexe, ont été rassemblés divers documents :

1. comment faire du SQL avec Access. Ce SGBD est très répandu sur les machines windows personnelles et il se trouve qu'il respecte une grande partie de la norme SQL. C'est l'outil idéal pour appréhender SQL chez soi.
2. comment installer Oracle sous une machine Linux ou Windows. C'est une bonne méthode pour apprendre à administrer Oracle mais elle est coûteuse en space disque, 1 Go environ et en performances. Une machine peu puissante est facilement écrasée par les ressources nécessaires à Oracle.
3. comment installer MySQL sous une machine Linux ou Windows. MySQL est une base de données moins complète mais beaucoup plus légère qu'Oracle. Contrairement à Access, ce SGBD peut être utilisé dans des applications réelles essentiellement sur des machines Linux.
4. comment faire du SQL avec l'outil Microsoft Query. Celui-ci permet de se connecter à quasiment toute base de données sous windows (Access, Oracle, MySQL,...).

1 L'environnement SQLPLUS d'Oracle

SQLPLUS est l'outil d'Oracle permettant l'utilisation du langage SQL. Cependant il offre en plus diverses commandes de manipulation de commandes SQL, de formatage des affichages écran etc ... formant ce que nous appellerons l'environnement SQLPLUS et qui est présenté partiellement dans ce chapitre.

SQLPLUS s'appelle à partir d'Unix par la commande :

sqlplus

Apparaît alors le message de connexion suivant :

```
SQL*Plus: Version 3.0.7.1.1 - Production on Thu Oct 10 13:24:03 1991

Copyright (c) Oracle Corporation 1979, 1989.  All rights reserved.

Enter user-name: serge
Enter password:
Connected to: ORACLE RDBMS V6.0.30.2.1, transaction processing option - Production
PL/SQL V1.0.30.0.1 - Production
```

Il vous est demandé votre nom ainsi que votre mot de passe. Consultez votre enseignant pour connaître ces deux informations. Pour vous connecter, vous pouvez aussi utiliser la syntaxe

```
sqlplus nom_utilisateur/mot_de_passe
```

Par exemple

```
sqlplus serge/serge
```

Une fois la connexion avec Oracle établie, SQLPLUS affiche son message d'attente :

```
SQL>
```

indiquant qu'il attend une commande SQL ou SQLPLUS :

Les commandes SQL permettent de créer, mettre à jour et exploiter les tables de données.

```
Ex : select * from biblio;
```

Les commandes SQLPLUS permettent de manipuler l'environnement dans lequel vont s'exécuter les commandes SQL :

a éditer, sauvegarder, récupérer des commandes SQL
b préciser le formatage désiré pour le résultat des requêtes SQL
c diverses commandes

```
Ex : describe biblio
```

1.1 Syntaxe des commandes SQL

Voici quelques règles d'écriture des commandes SQL :

1 Elles peuvent s'écrire indifféremment en majuscules ou minuscules. Par la suite, nous écrirons les noms des tables et colonnes en majuscules et le reste en minuscules.
2 Une commande SQL se termine par **;** ou **/** ou une ligne blanche :

`;`	indique la fin de la commande et demande son exécution
`/`	idem à ; mais doit être seul sur sa ligne.
`ligne blanche`	termine la commande sans lancer son exécution

3 Une commande SQL peut s'étaler sur plusieurs lignes. Après chaque ligne, l'interpréteur génère une ligne supplémentaire numérotée et ce tant qu'il n'a pas rencontré la fin de la commande.

```
a select * from biblio;
b select *
  2 from biblio;  <--- 2 est le N° de ligne
```

sont deux commandes identiques.

1.2 Syntaxe des commandes SQLPLUS

Voici quelques règles d'écriture des commandes SQLPLUS :

a La commande peut être entrée indifféremment en majuscules ou minuscules.
b La plupart des commandes SQLPLUS ont une abbréviation. Par exemple la commande **input** peut être abrégée par i.
c Une commande SQLPLUS peut être tapée sur plusieurs lignes, chaque ligne intermédiaire étant terminée par **-** . SQLPLUS commence la ligne suivante par > :

```
SQL> column genre -
> heading 'GENRE DU LIVRE'
```

d Une commande SQLPLUS ne se termine par rien de particulier. Cependant le point-virgule est accepté.

1.3 Quelques commandes SQLPLUS

Nous nous proposons ici de voir quelques commandes SQLPLUS qui nous seront utiles dans notre étude du langage SQL.

1.3.1 Sortie de SQLPLUS

syntaxe **exit**
action ramène au système d'exploitation

1.4 Exécuter une commande système

syntaxe **host commande_système**

action exécute la commande du système d'exploitation.

syntaxe **host**

action fait apparaître le "prompt" du système d'exploitation. On peut alors taper des commandes quelconques. On revient à SQLPLUS par la commande exit.

Exemples :

```
SQL> host pwd <-- répertoire courant ?
/users/serge/oracle/sqlplus <-- résultat

SQL> host ll <-- contenu du répertoire courant ?
total 0<-- rien

SQL> host >fic<-- on crée un fichier vide

SQL> host ll <-- vérification
total 0
-rw-rw-r--   1 serge    enseign        0 Oct 11 15:14 fic

SQL> host mkdir rep <-- on crée un répertoire

SQL> host ll <-- vérification
total 1
-rw-rw-r--   1 serge    enseign        0 Oct 11 15:14 fic
drwxrwxr-x   2 serge    enseign      512 Oct 11 15:15 rep


SQL> host cd rep  <-- on change de répertoire courant


SQL> host pwd <-- vérification
/users/serge/oracle/sqlplus <-- ça n'a pas marché



SQL> host ll <-- vérification : le répertoire courant n'a effectivement pas changé
total 1
-rw-rw-r--   1 serge    enseign        0 Oct 11 15:14 fic
drwxrwxr-x   2 serge    enseign      512 Oct 11 15:15 rep
```

On remarque qu'on ne peut changer de répertoire courant par la commande **host**. Essayons l'autre méthode :

```
SQL> host  <-- on appelle le système

$ pwd<-- on est sous le système. Répertoire courant ?
/users/serge/oracle/sqlplus


$ mkdir rep  <-- on crée un répertoire

$ ll <-- vérification
total 1
drwxrwxr-x   2 serge    enseign      512 Oct 11 15:25 rep

$ cd rep <-- changement de répertoire courant

$ pwd<-- vérification
/users/serge/oracle/sqlplus/rep <-- ça a marché

$ exit <-- retour à SQLPLUS

SQL> host pwd <-- répertoire courant ?
/users/serge/oracle/sqlplus <-- ce n'est plus rep mais SQLPLUS de nouveau
```

Les deux exemples précédents montrent que le répertoire courant pour SQLPLUS est celui à partir duquel il a été lancé. Il ne semble pas possible d'en changer. Cette notion de répertoire courant est importante car c'est là que SQLPLUS rangera certains des fichiers qu'il produira.

1.4.1 Gestion du buffer SQL

Sous SQLPLUS, on entre des commandes SQL ou SQLPLUS. La dernière commande SQL entrée au clavier est enregistrée dans une zone appelée buffer SQL. Tant qu'elle est présente dans ce buffer, la commande peut être modifiée, sauvegardée, relancée, etc... Les commandes de gestion du buffer SQL sont des commandes SQLPLUS et obéissent donc à la syntaxe déjà présentée. Noter que les commandes SQLPLUS émises ne sont pas mémorisées.

1.4.1.1 Edition du buffer

Les commandes d'édition du buffer s'appliquent à une seule des lignes constituant la commande SQL qui y est contenue. Celle-ci est signalée par une étoile et est appelée ligne courante.

Exemple :

```
SQL> select * <-- commande SQL sur 3 lignes
  2  from biblio  <-- elle est automatiquement enregistrée dans le buffer
  3  where prix>100;

SQL> list  <-- commande SQLPLUS visualisant le buffer SQL
  1  select *
  2  from biblio
  3* where prix>100 <-- la ligne 3 est ligne courante

SQL> list 2  <-- on demande à voir la ligne n°2 du buffer
  2* from biblio  <-- elle est devenue ligne courante
```

Voici les commandes d'édition du buffer SQL :

Commande	Abbréviation	But
`APPEND texte`	A texte	ajoute texte à la fin de la ligne courante
`CHANGE /ancien/nouveau/`	C /ancien/nouveau/	change texte ancien en texte nouveau dans la ligne courante
`CHANGE /texte`	C /texte	supprime texte dans la ligne courante
`DEL`		supprime la ligne courante
`INPUT`	I	entre en saisie de lignes supplémentaires
`INPUT texte`	I texte	ajoute texte au buffer
`LIST`	L	visualise toutes les lignes
`LIST n`	L n	visualise la ligne n° n
`LIST *`	L *	visualise la ligne courante
`LIST LAST`	L LAST	visualise la dernière ligne
`LIST m n`	L m n	visualise les lignes m à n
`CLEAR BUFFER`	CL BUFF	vide le buffer

Exemples

```
SQL> list  <-- contenu du buffer SQL
  1  select *
  2  from biblio
  3* where prix>100

SQL> clear buffer <-- vide le buffer
buffer cleared

SQL> list  <-- vérification
No lines in SQL buffer.

SQL> input <-- ajoute des lignes au buffer
  1  select *
  2  from biblio
```

```
  3  where prix>100
  4 <-- on termine par une ligne blanche pour que la commande
  <-- ne soit pas exécutée

SQL> l <-- vérification (l=list)
  1  select *
  2  from biblio
  3* where prix>100

SQL> del <-- supprime la ligne courante (3 ici)

SQL> l <-- vérification
  1  select *
  2* from biblio

SQL> l 1 <-- visualise ligne 1 qui devient ligne courante
  1* select *

SQL> l 2 <-- visualise ligne 2
  2* from biblio

SQL> i <-- ajoute des lignes (i=input)
  3  where prix>100
  4

SQL> l <-- vérification
  1  select *
  2  from biblio
  3* where prix>100

SQL> c/100/200/ <-- change 100 en 200 dans la ligne courante (ligne 3 ci-dessus)
  3* where prix>200 <-- résultat

SQL> l 2 <-- ligne 2 devient ligne courante
  2* from biblio

SQL> a 2 <-- ajoute 2 en fin de ligne courante (a=append)
  2* from biblio2 <-- résultat

SQL> l
  1  select *
  2  from biblio2
  3* where prix>200
```

Une autre manière d'éditer le buffer SQL est d'utiliser un éditeur de texte par la commande EDIT. Celle-ci appelle l'éditeur dont le nom est défini par la variable système _EDITOR. On peut obtenir la liste de ces variables par la commande DEFINE :

```
SQL> define  <-- liste des variables définies

DEFINE _EDITOR= "vi" (CHAR) <-- l'éditeur est ici vi.
DEFINE _O_VERSION =  "ORACLE  RDBMS  V6.0.30.2.1,  transaction  processing  option  -  Production  PL/SQL
V1.0.30.0.1 - Production" (CHAR)
DEFINE _O_RELEASE = "6003002" (CHAR)
```

Dans l'exemple précédent, la commande EDIT copie le buffer dans un fichier appelé afiedt.buf du répertoire courant puis appelle l'éditeur vi pour éditer ce fichier. On modifie et sauvegarde le fichier par les commandes habituelles de l'éditeur *vi*. Il sera recopié dans le buffer SQL.

```
SQL> l <-- liste le buffer
  1  select *
  2  from biblio2
  3* where prix>200

SQL> edit  <-- édition du buffer avec vi
       // changer 200 en 100
Wrote file afiedt.buf<-- création du fichier afiedt.buf

SQL> host ll <-- vérification
total 1
-rw-rw-r--   1 serge    enseign       38 Oct 11 15:35 afiedt.buf

SQL> host cat afiedt.buf <-- contenu de afiedt.buf
select *
from biblio
where prix>100
/

SQL> l <-- contenu du nouveau buffer
  1  select *
  2  from biblio
  3* where prix>100
```

Ceci est un exemple, cliquez sur le lien de téléchargement pour obtenir le cours complet.