# SQL/PLUS

# SOMMAIRE

## *1. - INTRODUCTION*

SQL*PLUS est un environnement de travail permettant :
**->** De saisir en conversationnel une commande SQL
**->** De modifier une commande saisie
**->** De lancer son exécution
**->** De la stocker dans un fichier puis de la rappeler
**->** De définir des paramètres de mise en forme permettant de contrôler la mise en page du résultat d'un select
**->** SQL*PLUS est disponible à partir de la version 5 d'ORACLE

## *2. - APPEL ET SORTIE DE SQL*PLUS*

## 2.1. - APPEL DE SQL*PLUS

### Syntaxe 1 :

SQLPLUS [nom_utilisateur][/mot_de_passe][@base]

Si l'on ne spécifie pas le nom d'utilisateur ou de mot de passe, SQL*PLUS demande le nom et/ou le mot de passe.
Le mot de passe est alors saisi en mode **'no echo'**.

Le paramètre **'@base'** permet de spécifier le nom de la base à laquelle l'utilisateur désire se connecter, en cas de connexion à une autre base que celle par défaut. La base demandée doit être accessible par SQL*NET.

@ est le signe de connexion à une base distante.
Base = nom logique équivalent à "Initiale Protocole réseau : n. serveur : n base

*Exemples :*

*SQLPLUS cours1/cours1*
*SQLPLUS cours2/cours2@dirregion*
*SQLPLUS courA/courA@ t : REG 1 : Dirregion*

### Syntaxe 2 :

SQLPLUS @nom_de_fichier
SQLPLUS exécutera les commandes contenues dans le fichier

Le premier enregistrement du fichier doit contenir le nom d'utilisateur ORACLE suivi éventuellement de **'/'** et du mot de passe. En cas d'absence de celui-ci, SQL*PLUS le demandera avant exécution du fichier.

*Exemples :*

*SQLPLUS @toto*

### Syntaxe 3 :

SQLPLUS/NOLOG
SQL*PLUS est lancé sans connexion à la base.

### Syntaxe 4 :

SQL*PLUS

Cette commande permet d'obtenir le numéro de version de SQL*PLUS

***Remarques :***

***Au démarrage de la session SQL*PLUS, le fichier LOGIN.SQL est lancé automatiquement, s'il existe dans le répertoire courant.***

## 2.2. - CONNEXION, DECONNEXION D'UNE BASE

### 2.2.1. - CONNEXION A UNE BASE

**Syntaxe :**

CONNECT nom_utilisateur[/mot_de_passe][@base]

Si l'on ne spécifie pas de mot de passe, SQL*PLUS le demande. Le mot de passe est alors saisi en mode **'no echo'.**

Le paramètre **'@base'** permet de spécifier le nom de la base à laquelle l'utilisateur désire se connecter, en cas de connexion à une autre base que celle par défaut. La base demandée doit être accessible par SQL*NET.

Cette commande permet également de se reconnecter sous un autre compte ORACLE au cours d'une session SQL*PLUS.

*Exemples :*

*CONNECT cours1/cours1*
*CONNECT cours2/cours2@dirregion*

### 2.2.2. - DECONNEXION D'UNE BASE

**Syntaxe :**

DISCONNECT

L'utilisateur n'est plus connecté à la base.

## 2.3. - Sortie de SQL*PLUS

**Syntaxe :**

EXIT ou QUIT

Remarques : commit implicite exécuté lorsque l'on fait un EXIT ou QUIT

# *3. - L'EDITEUR SQL*PLUS*

## 3.1. - AJOUT DE LIGNES

La commande **APPEND** (abréviation **A**) permet d'ajouter du texte en fin de la ligne courante.

**Syntaxe :**

A texte

L'espace entre la commande et le texte est facultatif si le texte ne commence pas par une lettre. Il n'est pas considéré comme faisant partie du texte. Par contre, un deuxième blanc est considéré comme faisant partie du texte et est donc inséré dans la ligne courante.

*Exemple :*

```
SQL>SELECT ref, nom
      FROM abo
      WHERE dnais > '01-JAN-70'

SQL>L1                    --> Positionnement sur la ligne 1

SQL>A, pnom

SQL> SELECT ref, nom, pnom
```

Remarque:
Pour se positionner sur la ligne 1, on peut également ne saisir que 1.

## 3.2. - INSERTION DE LIGNES

La commande **INPUT** (abréviation **I**) permet d'insérer une ou plusieurs lignes après la ligne courante.

**Syntaxe :**

**I**

L'insertion se termine à la fin de la ligne. Une seule ligne est insérée après la ligne courante.

**Syntaxe :**

**I** *[texte]*

La commande **I** place le buffer en mode insertion. Il est alors possible de saisir plusieurs lignes. L'insertion se termine par la saisie d'une ligne vide.

*Exemple :*

```
SQL>SELECT ref, nom
        FROM abo
        WHERE dnais > '01-JN-70'

SQL>L3                          --> Positionnement sur la ligne 3

SQL>I AND pnom like '%A%' or '%B%'

ou

SQL>I
        4 And pnom like '%A%'
        5
SQL>
```

## 3.3. - SUPPRESSION DE LIGNES

La commande **DEL** permet de supprimer la ligne courante.

**Syntaxe :**

**DEL**

La ligne courante est supprimée.

*Exemple :*

```
SQL>SELECT ref, nom
        FROM abo
        WHERE dnais > '01-JAN-70'
        AND pnom like '%A%' or '%B%'

SQL>L4                          --> Positionnement sur la ligne 4

SQL>DEL
```

**Attention**

**Le buffer se repositionne toujours sur la dernière ligne.**

```
SQL>SELECT ref, nom,
                pnom
        FROM abo
        WHERE ...
SQL>L2
SQL>del
SQL>L
SQL>SELECT ref, nom,
        FROM abo
        WHERE ...
SQL>del
SQL>L
SQL>SELECT ref, nom,
        FROM abo
```

## 3.4. - LA MODIFICATION D'UNE LIGNE

La commande **CHANGE** (abréviation **C**) permet de remplacer une chaîne de caractères par une autre chaîne sur la ligne courante.

### Syntaxe :

C/chaîne_1/chaîne_2/

Sur la ligne courante, la première occurrence de la chaîne_1 est remplacée par la chaîne_2.

Si la chaîne_2 est vide ou absente, alors la première occurrence de la chaîne_1 est supprimée.

Le séparateur '/' peut être remplacé par n'importe quel autre caractère. Le dernier séparateur n'est pas nécessaire, sauf pour insérer des blancs en fin de la chaîne_2.

*Exemple :*

```
SQL > SELECT ref, nom
      FROM abo
      WHERE dnais > '01-JAN-70'
      AND pnom like '%A%' or '%B%'

SQL>L4                    --> Positionnement sur la ligne 4

SQL>C : B% : C :
```

**<u>Note</u> : Convention méta-caractères classiques : / ou :**
**Si chaîne_1 est reproduite 2 fois dans la ligne, il ne remplace que la première**

## 3.5. - LA COMMANDE LIST

La commande **LIST** (abrévitaion **L**) permet de lister une ou plusieurs lignes du buffer de travail.

**Syntaxe 1 :**

**L**

Permet de lister l'ensemble des lignes du buffer de travail, la dernière ligne devient ligne courante.

**Syntaxe 2 :**

**L n**

Permet de lister la n-ième ligne du buffer de travail, celle-ci devient ligne courante.

**Syntaxe 3 :**

**L n m**

Permet de lister la n-ième à la m-ième ligne du buffer de travail, cette dernière devient ligne courante.

**Syntaxe 4 :**

**L ***

Permet de lister la ligne courante du buffer de travail.

*Exemple :*

```
SQL > SELECT ref, nom
      FROM abo
      WHERE dnais > '01-JAN-70'
      AND pnom like '%A%' or '%B%'

SQL>L3                        --> Positionnement sur la ligne 3

3* WHERE dnais > '01-JAN-70'
```

## 3.6. - LA FIN D'UNE COMMANDE

SQL*PLUS distingue une commande SQL d'une commande SQL*PLUS en analysant le premier élément de la commande.

**S'il s'agit d'un verbe SQL :**

**->** SQL*PLUS attend des lignes suites.

**Si la commande SQL se termine sur :**

**->** Une ligne vide., la commande est stockée dans le buffer de travail.

**->** Un **';'** en fin de ligne ou un **'/'** en début de ligne, dans ce cas, la commande est stockée dans le buffer de travail et est exécutée.

**S'il s'agit d'un verbe SQL*PLUS :**

**->** SQL*PLUS exécute immédiatement l'ordre sans le stocker dans le buffer de travail.

## 3.7. - L'EXECUTION D'UNE COMMANDE

Une commande SQL contenue dans le buffer de travail peut être exécutée ou réexécutée par :

**->** La commande **RUN** (abréviation **R**) qui liste l'ordre SQL et lance son éxécution.

**->** La commande **'/'** qui lance l'exécution sans lister l'ordre SQL.

**->** La saisie de **';'** en fin de ligne ou **'/'** en début de ligne lors de la saisie de l'ordre SQL.

## 3.8. - LES BUFFERS DE TRAVAIL

On ne peut pas construire dans le buffer SQL un fichier de commandes contenant à la fois des commandes SQL et des commandes SQL*PLUS.

Pour lever cette limitation, SQL*PLUS permet de se créer des buffers de travail dans lesquels on pourra construire des procédures contenant à la fois des ordres SQL*PLUS et SQL.

Les commandes contenues dans ces buffers ne sont pas directement exécutées par SQL*PLUS. Par contre, on peut sauvegarder le contenu de ces buffers par la commande SAVE puis l'exécuter par la commande START( ou @)

### 3.8.1. - LE BUFFER COURANT

Un seul buffer est courant à un instant donné. Toutes les commandes à l'exception de **'/'** s'applique au buffer courant.

Par défaut, le buffer courant est le buffer SQL.

### 3.8.2. - DEFINITION DU BUFFER COURANT

Pour travailler dans un buffer autre que le buffer SQL, il faut définir un buffer nommé.

**Syntaxe :**

SET BUFFER nom

Le buffer est créé s'il n'existait pas et devient courant.

Le nom du buffer courant est stocké dans la variable BUFFER et peut donc être obtenu par la commande SHOW BUFFER

### 3.8.3. - RETOUR AU BUFFER SQL

Pour revenir au buffer SQL, il suffit de le remettre buffer courant.

**Syntaxe :**

SET BUFFER SQL

**Remarques :**

La saisie d'un ordre SELECT, START ou RUN sous le prompt SQL*PLUS ramène au buffer SQL.

La saisie d'une commande **'/'** sous le prompt SQL*PLUS ainsi qu'en mode insertion provoque le retour au buffer SQL avec exécution de celui-ci.

### 3.8.4. - EFFACEMENT DU CONTENU D'UN BUFFER

L'instruction **CLEAR BUFFER** permet d'effacer le contenu du buffer courant.

**Syntaxe :**

CLEAR BUFFER

### 3.8.5. - LES COMMENTAIRES

L'instruction SQL*PLUS **REM** permet d'introduire des commentaires dans un fichier de commande.

**Syntaxe :**

REM texte

***ATTENTION :*** Au cas où le dernier caractère du texte de commentaire est celui de continuation, alors la ligne suivante est considérée comme faisant partie du commentaire.

## 3.9. - APPEL D'UN EDITEUR EXTERIEUR

La commande SQL*PLUS **ED** appelle l'éditeur déclaré de la machine hôte en lui passant le contenu du buffer SQL.

L'éditeur appelé est défini par la commande **DEF_EDITOR.**

**Syntaxe :**

DEF_EDITOR = "commande"

Remarque:
‘’ commande ’’ correspond au chemin de commande

*Exemple :*

*DEF_EDITOR = "xedit"*

## *4. - L'AIDE EN LIGNE*

La commande **HELP** permet d'obtenir l'aide en ligne concernant les ordres SQL et SQL*PLUS il est cependant nécessaire que celle-ci soit chargée lors de l'installation d'ORACLE.

**Syntaxe :**

HELP [nom_commande]

Si le nom de commande est saisie, l'utilisateur obtient l'aide concernant celle-ci. Dans le cas contraire, il obtient la liste des commandes pour lesquelles une aide est disponible.

# *5. - APPEL AU SYSTEME D'EXPLOITATION*

La commande **HOST** permet l'exécution d'une commande du système d'exploitation de la machine hôte. En fin d'exécution, il y a retour à SQL*PLUS.

**Syntaxe :**

HOST [COMMANDE]

En cas de non saisie d'une commande, l'utilisateur se retrouve sous l'analyseur de commande de la machine hôte.

La commande '$' ou '!' (selon la machine) est équivalent à la commande HOST.

Exemple :

HOST [DIR C:]

# *6. - LES FICHIERS DE COMMANDES*

SQL*PLUS permet d'exécuter des commandes SQL et SQL*PLUS stockées dans un fichier.

## 6.1. - LA SAUVEGARDE DU BUFFER DE TRAVAIL

La commande **SAVE** permet de sauvegarder le contenu du buffer courant dans un fichier.

**Syntaxe :**

SAVE nom_fichier [APPEND] [CREATE] [REPLACE]

Le type par défaut du fichier est **'SQL'.**

L'option **APPEND** permet d'ajouter le contenu du buffer courant à la fin du fichier spécifié.

L'option **CREATE** empêche l'écrasement du fichier au cas où celui-ci existe déjà.

L'option **REPLACE** écrase le contenu du buffer, s'il existait (option par défaut).

La commande **SAVE** termine la transaction en cours, un **COMMIT** est implicitement effectué.

## 6.2. - LE RAPPEL D'UN FICHIER D'ORDRE SQL*PLUS

### 6.2.1. - LE RAPPEL SANS EXECUTION

La commande **GET** permet de charger un fichier d'ordre dans le buffer courant.

**Syntaxe :**

GET nom_fichier [NOLIST]

Le type par défaut du fichier est **'SQL'**

L'option **NOLIST** évite l'affichage du fichier lors du chargement.

### 6.2.2. - LE RAPPEL AVEC EXECUTION

Les commandes **START** et **'@'** permettent d'exécuter un fichier d'ordre SQL et SQL*PLUS.

**Syntaxe 1 :**

START nom_fichier [param_1 [param_2 ...]]

Le type par défaut du fichier est **'SQL'.**

Le fichier peut contenir plusieurs ordres SQL. Ceux-ci doivent se terminer par **';'** ou être séparés par une ligne contenant le seul caractère **'/'**.

Le dernier ordre SQL exécuté est conservé dans le buffer courant.

Il est possible de passer directement des valeurs pour les paramètres de substitution éventuellement contenus dans le fichier. Dans ce cas, la première valeur est associée à la variable de substitution &1, la deuxième à &2 et ainsi de suite.

**Syntaxe 2 :**

@ nom _fichier

Syntaxe comparable à l'ordre START.

### 6.2.3. - CONTROLE DE L'EXECUTION

Lors de l'exécution d'une commande ou d'un fichier de commande, il est possible de contrôler, de tracer les commandes exécutées.

**Syntaxe :**

SET ECHO [ON/OFF]

ON permet de contrôler l'exécution.

Exemple :

SQL> select * from emp where job='PRESIDENT'
SQL> set echo off
SQL> ED test
- Apparition du traitement de texte associé

SQL> start test
- Si echo off : SQL *plus n'affiche pas le texte de la requête
- Si echo on : SQL *plus affiche en plus du résultat, le texte de la requête.

## *7. - LES PARAMETRES*

SQL*PLUS permet d'introduire dans une commande SQL ou SQL*PLUS des paramètres dont la valeur est fournie lors de l'exécution.

Ces paramètres sont désignés par un nom ou un numéro, précédé d'un ou de deux caractères **'&'.**

### 7.1. - LES PARAMETRES PERMANENTS

Si le nom du paramètre est précédé d'un seul **'&'**, la valeur donnée en réponse à la question **'enter value for :'** sera associée au paramètre de façon éphémère. SQL*PLUS demandera une valeur pour chaque occurrence du paramètre.

Si le nom du paramètre est précédé de deux **'&'**, la valeur donnée en réponse à la question **'enter value for :'** sera accessible de façon permanente. SQL*PLUS ne demandera qu'une seule fois la valeur du paramètre, même si celui-ci est présent plusieurs fois.

## 7.2. - Définition de paramètre

La commande **DEF** permet de définir des paramètres.

**Syntaxe 1 :**

DEF nom_paramètre = valeur

La valeur affectée est associée au paramètre pendant toute la session SQL*PLUS, sauf en cas de redéfinition de ce même paramètre ou d'annulation au moyen de la commande **UNDEF**.

**Syntaxe 2 :**

DEF nom_paramètre

Permet de connaître la valeur affectée au paramètre.

**Syntaxe 3 :**

DEF

Permet de connaître la valeur affectée à tous les paramètres.

La commade **UNDEF** permet d'annuler un paramètre.

**Syntaxe :**

UNDEF nom_paramètre

### 7.3. - LA COMMANDE ACCEPT

La commande **ACCEPT** force SQL*PLUS à attendre la saisie d'un paramètre.

Avant de se mettre en attente, SQL*PLUS doit afficher un message informant qu'une valeur pour un paramètre doit être saisie. Ceci peut se faire au moyen de la commande **PROMPT** qui permet d'afficher un texte (La commande **PROMPT** peut être sur une ligne distincte ou être incluse dans la commande **ACCEPT**).

**Syntaxe :**

[PROMPT]
ACCEPT nom_paramètre [PROMT texte] [HIDE]

L'option HIDE supprime l'écho lors de la saisie de la réponse.

*Exemple :*

```
ACCEPT   num_ref PROMPT entrer un numéro d'abonné
TTITLE    "Abonné référencé et num_ref"
SELECT   *
FROM     abo
WHERE   ref = &num_ref;
```

## *8. - LA SORTIE VERS UN FICHIER SPOOL*

### 8.1. - LA MISE EN SERVICE

La commande **SPOOL** permet de copier l'équivalent du dialogue dans un fichier.

**Syntaxe 1 :**

SPOOL nom_fichier

Le type par défaut est **'LST'** sur machine MS-DOS, UNIX ... et 'LIS' sur machine VAX/VMS.

Si le fichier n'existe pas il est créé, dans le cas contraire il est supprimé et recréé.

**Synthaxe 2 :**

SPOOL

Permet de connaître le nom du fichier spool.

## 8.2. - ARRET SANS IMPRESSION

La commande **SPOOL OFF** permet de stopper la copie du dialogue et ferme le fichier spool.

**Syntaxe :**

SPOOL OFF

## 8.3. - ARRET AVEC IMPRESSION

La commande **SPOOL OUT** permet de stopper la copie du dialogue, ferme le fichier spool et demande l'impression du fichier spool.

**Syntaxe :**

SPOOL OUT

# *9. - LA MISE EN FORME DES RESULTATS*

L'environnement SQL*PLUS contient un certain nombre de paramètres que l'utilisateur peut modifier, et qui déterminent la façon dont sont présentés les résultats d'un SELECT.

## 9.1. - FORMATAGE DES COLONNES

La commande de **COLUMN** (abréviation **COL**) permet d'associer à un nom de colonne différents paramètres de mise en forme.
Ces options s'appliquent à toutes les colonnes dont les noms sont identiques à ceux cités dans la commande COL, et ceci pour toute la durée de la session SQL*PLUS.

**Syntaxe 1 :**

COL nom_colonne option_1 [option_2 ...]

Cette commande permet d'associer une ou plusieurs options à une colonne.

**Syntaxe 2 :**

COL nom_colonne

Cette commande permet de lister les différentes options associées à une colonne.

**Syntaxe 3 :**

COL

Cette commande permet de lister les différentes options associées à toutes les colonnes.

CLEAR COL supprime les caractéristiques de toutes les colonnes

**Liste des options :**

**FORMAT masque :**

Détermine le format de mise en forme du contenu de la colonne.
Format caractère : **An** (**n** défini le nombre de caractères de la colonne).
Format numérique : **9 (n fois)**

**HEADING Chaîne :**

Définit le titre de la colonne.
Si le titre contient des blancs, il doit être entre **(')** ou **(").**
Le titre peut être sur plusieurs lignes, dans ce cas le caractère "**|"** indique le passage à la ligne.

Défaut : Le titre est le nom de la colonne.

**JUSTIFY { L[EFT | C[CENTER] | R[IGHT] } :**

Définit le cadrage à gauche, centré ou à droite du titre de la colonne.
Défaut : RIGHT pour les colonnes de type caractère
LEFT pour les colonnes de type numérique

**WRA[PPED] | WOR[D_WRAPPED] | TRU[NCATED] :**

Spécifie si le contenu de la colonne est tronqué ou bien continue sur la ligne suivante, dans le cas ou sa longueur dépasse la largeur de la colonne.
L'option WORD_WRAPPED évite qu'un mot soit coupé en deux.

Défaut : WRAPPED

**NEWLINE :**

Provoque le passage à la ligne suivante avant l'affichage de la colonne.

**NULL chaîne :**

Spécifie la chaîne de caractères représentant la valeur nulle.

Défaut : blanc

**PRINT |NOPRINT :**

Permet de supprimer ou remettre en service l'impression d'une colonne.

Défaut : PRINT

**ALIAS nom :**

Définit un synonyme pour une colonne.

**LIKE nom :**

Permet de recopier les attributs définis pour une autre colonne.

**TEMPORARY :**

Les options seront annulées après le prochain SELECT.

**CLEAR :**

Supprime toutes les options.

**DEFAULT :**

Les options non spécifiées sont mises à leur valeur par défaut.

**ON/OFF :**

OFF permet le retour aux options par défaut tout en conservant les options qui ont été définies.
ON remet en fonction les options définies.

## 9.2. - LES TITRES

Il est possible de définir des titres qui apparaissent en haut et en bas de chaque page.

**Syntaxe 1 :**

TTITLE texte

Le texte est centré en haut de chaque page. Il peut en être affiché sur plusieurs lignes (caractère **'|'** pour permettre le passage à la ligne).

En plus du titre, la date est affichée en haut à gauche de chaque page, les pages sont numérotées en haut à droite.

**Syntaxe 2 :**

BTITLE Texte

Identique à la syntaxe 1 mais pour le bas de page.

**Syntaxe 3 :**

TTITLE [COL n | SKIP n | TAB n | LEFT | CENTER | RIGHT | "texte"]

Les spécifications COL, SKIP, TAB, LEFT, CENTER, RIGHT définissent l'emplacement du texte. Chaque spécification peut se répéter.

**Syntaxe 4 :**

BTITLE [COL n | S KIP n | TAB n | LEFT | CENTER | RIGHT | "texte" ]

Identique à la syntaxe 3 mais pour le bas de page.

**Syntaxe 5 :**

TTITLE ON | OFF

OFF permet de supprimer l'affichage du titre de haut de page, tout en conservant sa description pour une nouvelle remise en service.

ON remet en service l'affichage du titre de haut de page.

**Syntaxe 6 :**

BTITLE ON | OFF

Identique à la syntaxe 5 mais pour le bas de page.

**Règle de syntaxe** :

Le | indique un saut de ligne.

Le - indique la continuation de la commande à la ligne suivante

Exemple:

Tiltle center ‘’ EMPLOYER ’’
SKIP CENTER ‘’PAR DEPARTEMENT’’

## 9.3. - LES RUPTURES

La commande **BREAK** permet de spécifier des actions à exécuter en cas de rupture sur un critère donné (colonne, ligne, page, report).

**Syntaxe :**

BREAK ON [nom_col | ROW | PAGE | REPORT] [SKIP n|PAGE] [DUP] [ON...]

La rupture peut être effectuée sur chaque changement de valeur d'une colonne **(nom_col)**, sur chaque ligne de table **(ROW)**, sur chaque page **(PAGE)** ou en fin d'état **(REPORT).**

L'option **SKIP** génère un saut de ligne lors de la rupture, **PAGE** génère un saut de page.

L'option **DUP** spécifie que toutes les valeurs seront affichées dans les colonnes définissant les ruptures. Par défaut, elles ne sont affichées que lors d'un changement de valeur **(NODUP par défaut)**.

Il est possible d'imbriquer plusieurs critères de rupture au sein d'une même commande **BREAK.**

La commande **CLEAR BREAK** permet d'annuler les ruptures définies.

Exemple:
BREAK ON deptno SKIP

**Syntaxe :**

CLEAR BREAK

### 9.4. - LES TOTALISATIONS

La commande **COMPUTE** permet d'obtenir la totalisation d'une colonne sur rupture pour un critère donné (colonne, ligne, page, report).

**Syntaxe :**

```
COMPUTE [SUM | COUNT | NUMBER | MIN | MAX | AVG | VAR | STD]
        OF  col_1   [col_2   ...]
        ON [nom_col_break_1 |ROW | PAGE | REPORT ]
```

Provoque l'affichage d'une ligne supplémentaire contenant un total de la colonne col_1 à chaque rupture définie par le critère de rupture.

Dans le cas d'une totalisation sur rupture d'une colonne, celle-ci doit être définie dans les critères de rupture de la clause **BREAK.**

CLEAR COMPUTE Annule les calculs définis

Exemple :
COMPUTE SUM OF sal ON deptno REPORT

## *10. - LA DESCRIPTION DES TABLES - DESC*

La commande **DESC** permet d'obtenir la description des différentes colonnes d'une table.

**Syntaxe :**

DESC nom_table

Nom_table peut correspondre à une table ou à une vue.

*Exemple :*

*SQL> desc abo*

| Name | Null? | Type |
|---|---|---|
| REF | NOT NULL | NUMBER (3) |
| NOM | NOT NULL | CHAR (30) |
| PNOM | NOT NULL | CHAR (30) |
| ADR | NOT NULL | CHAR (30) |
| DNAIS | NOT NULL | DATE |

# *11. - LA MESURE DES PERFORMANCES*

## 11.1. - MESURE SUR UNE REQUËTE

La variable SQL*PLUS **TIMING** permet la production d'une ligne d'information à la fin de chaque commande SQL. Cette ligne comporte les informations suivantes (environnement DIGITAL/VMS) :

**=> ELAPSED** : Le temps écoulé

**=>CPU** : Le temps CPU consommé par la tâche SQL*PLUS

**=> BUFIO** : Le nombre d'E/S logiques ou accès aux buffers

**=> DIRIO** : Les nombre d'E/S physiques ou accès au disque

**=> FAULTS** : Le nombre de défauts de page

**Syntaxe :**

SET TIMING ON | OFF

L'option ON permet d'activer la production de la ligne d'informations, OFF permet de la désactiver.

La ligne d'information est système-dépendant.

Pour un environnement UNIX
=> ELAPSED

## 11.2. - CUMUL DES MESURES

### 11.2.1. - CREATION D'UNE ZONE DE CUMUL

**Syntaxe :**

TIMING START nom

La zone de cumul 'nom' devient la zone de cumul courante. Toutes les informations précédentes sont cumulées dans cette zone
S'il y avait précédemment une zone de cumul en service, cette dernière est conservée en l'état mais son contenu n'évolue plus.

### 11.2.2. - AFFICHAGE D'UNE ZONE DE CUMUL

**Syntaxe :**

TIMING SHOW

Affiche le contenu de la zone de cumul courante.

### 11.2.3. - FIN DE CUMUL

**Syntaxe :**

TIMING STOP

Affiche le contenu de la zone de cumul courante, puis supprime cette zone de cumul.

## *12. - ORACLE* NET*

ORACLE*NET est une architecture permettant d'interconnecter des bases situées sur des machines différentes ou non, reliées par un réseau.

### 12.1. - CONNEXION A UNE BASE DISTANTE

SQL*PLUS permet de se connecter à une base distante, en spécifiant le chemin d'accès à la base. Celle-ci peut être effectuée lors de l'ouverture de la session SQL*PLUS **(SQLPLUS)** ou lors d'un ordre de connexion **(CONNECT)**

**Syntaxe 1 :**

SQLPLUS  nom_utilisateur[/mot_de _passe]@base*

**Syntaxe 2 :**

Lorsque l'on est déjà connecté à SQL*PLUS:

CONNECT nom_utilisateur[/mot_de_passe]@base*

La base distante spécifiée doit être accessible par le réseau ORACLE*NET.

*@Base = nom logique

Base remplace la chaîne de connexion à une base distante

Chaîne = [type réseau : [nom serveur] : nom de base

Type réseau = t -> TCP/IP (UNIX)
d -> DECNET (VAX/VMS)
i -> ISO/DSA (GCOS7)

*Exemple : t : REGI : DIRREGION*

## 11.2. - TRANSFERT DE DONNEES

La commande **COPY** permet de transférer des informations entre deux bases reliées par **ORACLE*NET.**
Du coté de la base source, un **SELECT** est effectué. Du coté de la base destinataire, les insertions sont effectuées après avoir éventuellement créé la table réceptrice.

**Syntaxe :**

```
COPY  [FROM   nom_utilisateur[/mot de passe]@base]
      [TO     nom_utilisateur [/mot depasse]@base]
      {APPEND | CREATE | REPLACE | INSERT} nom_table [(col_1, ...)]
      USING   ordre _select
```

La clause **FROM** désigne la base et l'utilisateur source. Par défaut, c'est la base et l'utilisateur sous lesquels on est connecté.

La clause **TO** désigne la base et l'utilisateur de destination. Par défaut, c'est la base et l'utilisateur sous lesquels on est connecté.

Les options **APPEND, CREATE, REPLACE, INSERT** précisent les règles d'insertion.

**APPEND** : Insertion des enregistrements dans la table. Si celle-ci n'existe pas, elle est créée.

**CREATE** : La table ne doit pas exister, celle-ci est créée et les enregistrements sont insérés.

**REPLACE** : Suppression du contenu de la table destinataire et insertion des enregistrements dans la table. Si celle-ci n'existe pas, elle est créée.

**INSERT** : Suppression du contenu de la table destinataire et insertion des enregistrements dans la table. Celle-ci doit exister préalablement.

Exemple:

```
COPY from scott/tiger@t:boston -
TO todd/fox@t:chicago -
INSERT emp -
USING SELECT * from emp
```

## *13. - LES VARIABLES SQL*PLUS*

Un certain nombre de paramètres de l'environnement SQL*PLUS peuvent être modifiés par l'utilisateur, au moyen de la commande **SET.**

**Syntaxe :**

SET variable valeur

L'utilisateur peut consulter les valeurs d'une ou plusieurs variables SQL*PLUS, ou même de l'ensemble, au moyen de la commande **SHOW.**

**Syntaxe :**

SHOW variable1 [variable2] | ALL

Exemple:

SET TIMING ON
SHOW TIMING => TIMING ON
SHOW ALL => SQL*PLus donne l'etat de toutes les variables.

## 13.1. - CONTROLE DU FORMAT DE SORTIE

**LINE[SIZE] n**
Spécifie le nombre de caractères par ligne à l'affichage ou dans le fichier de sortie **(spool)**.

Défaut : 80
Maximum : 999

**PAGE[SIZE] n**
spécifie le nombre de lignes par page à l'affichage ou dans le fichier de sortie **(spool).**
Une valeur 0 correspond à l'absence de saut de page.
Défaut : 14

**NUM[WIDTH] n**

Spéficie la largeur par défaut des colonnes numériques.

Défaut : 10

**NUMF[ORMAT] format**

Spéficie le format par défaut des colonnes numériques.

**NEWP[AGE]n**

Spécifie le nombre de lignes à sauter pour changer de page.
En cas de valeur 0, SQL*PLUS utilise le caractère saut de page.
Défaut : 1

**EMBEDDED [ ON|OFF ]**

ON indique qu'il n'y a pas de saut de page entre une commande SELECT et son résultat.

Défaut : OFF

**NULL chaîne**

Spécifie la chaîne de caractères représentant les valeurs nulles.

Défaut : blanc

**SPA[CE] n**
Spécifie le nombre d'espaces entre chaque colonne du résultat.

Défaut : 1

**TAB ON|OFF**

OFF indique à SQL*PLUS de ne pas utiliser de tabulation dans ses sorties.

Défaut : ON

**UND[ERLINE] caractère**

Spécifie le caractère utilisé pour souligner les titres des colonnes.

**WRAP ON|OFF**

OFF indique à SQL*PLUS de tronquer les lignes trop longues (les champs qui ne tiennent pas sur la ligne de résultat ne sont pas affichés).
ON provoque la continuation sur la ligne suivante.

Défaut : ON

## 13.2. - CONTROLE DES COMPTE-RENDUS

**TERM[OUT] ON | OFF**

OFF supprime tout affichage sur le terminal lors de l'exécution d'un fichier.

Défaut : ON

**ECHO ON | OFF**
ON provoque l'affichage des commandes exécutées lors de l'exécution d'un fichier.

Défaut : OFF

**FEED[BACK] ON | OFF**

Spécifie à partir de combien de lignes le nombre de lignes résultantes est affiché en fin d'un SELECT.

0 supprime l'affichage du nombre de lignes résultantes.

Défaut : 6

**SHOW[MODE] ON | OFF**

ON provoque l'affichage de l'ancienne et de la nouvelle valeur d'une variable SQL*PLUS lors d'une commande SET.

**HEA[DING] ON | OFF**

OFF supprime l'affichage des titres de colonnes.

Défaut : ON

**PAU[SE] ON |OFF**

ON provoque l'attente de la frappe de la touche <return> avant l'affichage de chaque page.

Défaut : OFF

**SQLN[UMBER] ON | OFF**

ON provoque l'affichage de la numérotation des lignes lorsqu'on est en insertion dans un buffer.

OFF remplace la numérotation par l'invite.

Défaut : OFF

**SQLP[ROMPT] chaîne**
Spécifie la chaîne utilisée pour l'invite SQL*PLUS.

Défaut : 'SQL>'

**TI[ME] ON | OFF**

ON provoque l'affichage de l'heure avec l'invite.

Défaut : OFF

**SQLPRE[FIX] caractère**

Spécifie le caractère préfixe permettant de faire exécuter immédiatement une commande SQL*PLUS pendant que l'on est en insertion dans un buffer, sans en affecter son contenu.

Défaut :" # "

**VER[IFY] ON | OFF**

ON provoque l'affichage des lignes de commande avant et après chaque substitution de paramètre.

Défaut : ON

**TIMING ON | OFF**

ON provoque l'affichage d'informations sur le temps écoulé, le nombre d'E/S après chaque requête SQL.

Défaut : OFF

## 13.3. - CONTROLE DE LA SYNTAXE

**CMDS[EP] {carartère | OFF}**

Spécifie le caractère séparant plusieurs commandes SQL*PLUS sur la même ligne.

**DEF[INE] caractère :**

Spécifie le caractère identifiant les variables de substitution.

Défaut : '&'

**HEADS[EP] caractère :**

Spécifie le caractère indiquant le passage à la ligne suivante dans les titres.

Défaut : '|'

**SCAN ON |OFF :**

OFF supprime le traitement des variables de substitution.

Défaut : ON

**SQLT[ERMINATOR] caractère :**

Spécifie le caractère indiquant la fin de saisie d'une commande SQL.

Défaut : ';'

**SQLC[ONTINUE] caractère :**

Spécifie le caractère de continuation d'une commande SQL*PLUS.

Défaut : '-'

**SUFFIX chaîne :**

Spécifie le type par défaut des fichiers de commande SQL*PLUS.

Défaut : 'SQL'

## 13.4. - CONTROLE DES TRANSACTIONS

**AUTO[COMMIT] ALL |OFF |IMM**

**IMM** provoque un **COMMIT** après chaque ordre **SQL**.

**ALL** et **OFF** annule le commit automatique.

Défaut : ALL

## 13.5. - CONTROLE DES ZONES DE TRAVAIL

**BUFFER nom | SQL :**

Spécifie le buffer courant.

**WORK[SIZE]n :**

Spécifie la taille en Koctet (3 .à. 128) du contexte de travail (zone de travail nécessaire à l'exécution d'un ordre SQL).

0 signifie que la taille est celle définie au lancement d'ORACLE (fichier INIT.ORA).

Défaut : 0

**LONG n :**

Spécifie la taille de la zone de lecture des champs de type LONG (Les champs LONG sont tronqués à cette longueur).

Défaut : 80

**MAXD[ATA] n :**

Spécifie la taille maximale d'une ligne de table que SQL*PLUS peut traiter.

**ARRAY[SIZE] n :**

Spécifie le nombre de lignes de tables lues par SQL*PLUS à chaque appel au noyau d'ORACLE.

**COPYC[OMMIT] n :**

Spécifie le nombre de groupes de lignes insérés avant chaque COMMIT lors d'un COPY. Un groupe de lignes est le nombre de lignes défini par la variable ARRAYSIZE.
0 signifie un seul COMMIT à la fin du COPY.