SOMMAIRE

1 Caractéristiques principales

A l'origine **SEQUEL** : **S**tructured **E**nglish **QUE**ry **L**anguage (IBM 1970)

- Non procédural,
- Interface unique de gestion de la base,
- Normalisé,
- Utilisable par les "non informaticiens",

1.1 L'algèbre relationnelle

Opération	Description
Restriction	extrait, d'une ou plusieurs tables, les lignes qui satisfont certains critères
Projection	extrait des colonnes particulières d'une ou plusieurs tables
Jointure	"rapproche" 2 ou n tables selon des critères qui "associent" les lignes des 2 ou n tables
Union	combine 2 ou n tables pour donner toutes les lignes appartenant à l'une **ou** l'autre des tables
Différence	combine 2 tables pour donner toutes les lignes appartenant à l'une sans appartenir à l'autre
Intersection	combine 2 tables pour donner toutes les lignes appartenant à l'une **et** à l'autre

1.2 Exemples d'objets du SGBD ORACLE

- ***concernant les DONNEES*** :

Table, View, Synonym, Index, Sequence, ...

- ***concernant les TRAITEMENTS*** :

PROCEDURE, FUNCTION	instructions SQL et PL/SQL stockées dans la base sous forme "pseudo compilée" sans ou avec retour de valeur
PACKAGE	unité de stockage de procédures, fonctions, variables ...
TRIGGER	procédures déclenchées automatiquement par le noyau ORACLE si un évènement particulier se produit

- ***concernant la CONFIDENTIALITE*** :

USER	identifie un utilisateur reconnu par à la base
ROLE	ensemble de privilèges définis sur différents objets regroupés sous un nom

...

1.3 Le langage de définition des données

Verbes de définition d'objets **CREATE, ALTER, DROP, ...**

1.4 Le langage de manipulation des données

Verbes de manipulation de données : **SELECT, INSERT, UPDATE, DELETE, ...**

1.5 Le dictionnaire des données

Le dictionnaire des données regroupe toutes les caractéristiques de tous les objets créés dans la base.

Le dictionnaire est

- organisé en tables ORACLE,
- non optionnel,
- dynamique,
- interrogeable par SQL,

Accès total pour un DBA, accès partiel, suivant autorisation, pour un non DBA.

1.6 La confidentialité

- Accès à la base

Utilisateur : personne physique
User : objet reconnu par Oracle comme ayant des droits

Un user a le droit de :
- se connecter à la base, d'écrire dans la base, de créer ses objets
- de manipuler des objets dont il n'est pas propriétaire

- Privilèges particuliers

Privilèges de réaliser une action particulière limitée à un type d'objet particulier.
Ils peuvent être attribués à des utilisateurs ou à des rôles et servent à définir des types d'utilisateurs particuliers (administrateur, développeur, utilisateur).

- Privilèges sur les objets

Exemples d'objets : vue, table, séquence, rôle, procédure, fonction, package, ...

Le créateur d'un objet est propriétaire de l'objet et a tous les privilèges sur cet objet.

- Les vues

Une vue est un objet *(qui ressemble à une table mais qui n'est pas une table)* dérivé d'autres tables par l'application d'une requête.

(indépendance logique, commodité, confidentialité, intégrité)

1.7 Les contraintes d'intégrité

C’est un ensemble de conditions qui doivent être vérifiées par une ou plusieurs tables de la base, afin que les informations soient considérées comme cohérentes.

contrainte de type d'un attribut	CHAR, VARCHAR2, LONG, NUMBER, RAW, LONG RAW, DATE
contrainte d'attribut obligatoire	NOT NULL
contrainte de longueur maximum	VARCHAR2(12) ou NUMBER(10,2)
contrainte de valeur par défaut	DEFAULT
contrainte d'identifiant	PRIMARY KEY
contrainte d'unicité	UNIQUE KEY
contrainte d'intégrité référentielle	FOREIGN KEY ... REFERENCES ... (ON DELETE CASCADE)
contrainte d’attributs	CHECK

1.8 La transaction et les accès concurrents

Une transaction est un ensemble de mises à jour de la base dépendantes les unes des autres, ***effectuées complètement ou pas du tout.***

- **débit d'un compte**

```
update compte set solde = solde - 3000
where nocompte = 123456789 ;
```

- **crédit d'un compte**

```
update compte set solde = solde + 3000
where nocompte = 987654321 ;
```

Les accès concurrents :

Il y a accès concurrent quand plusieurs utilisateurs tentent de lire et/ou d’écriture sur les mêmes tables de la base en même temps.

En lecture : image avant sur données en cours de mise à jour,
image après sur données des transactions terminées

En écriture : (par défaut) partage des tables au niveau ligne.

La libération des verrous se fait en fin de transaction (commit, rollback, exit, ordre du langage de définition des données).

1.9 L'optimisation

Les index sont des structures particulières permettant un accès plus performant aux données.

1.10 L'administration de la base de données

Dimensionnement, configuration et création de la base physique - Modification de la base physique - Reprise à froid - Démarrage / Arrêt de la base - Définition des paramètres globaux - Création de nouveaux utilisateurs et de nouveaux profils - Attribution de privilèges système - Audit global - Consultation des tables du dictionnaire - Mise en place des sauvegardes physiques - Reprise après panne - Surveillance en temps réel de la base - Réorganisation physique de la base - Création d'autre DBA - Création du schéma relationnel : tables, vues, contraintes d'intégrité - Modification du schéma relationnel - Chargement des données - Création d'accélérateurs - Définition des rôles - Accord des droits aux utilisateurs - Mise en place des sauvegardes logiques - Audit des tables …

2 SELECT, syntaxe de base.

Catégories	Verbes SQL	Remarques
DRL : Data Retrieval Language - Langage de "récupération" des données.	**Select**	commande permettant de rechercher les données d'une ou plusieurs tables de la base.
DML : Data Manipulation Language - Langage de manipulation des données.	**Insert**, **Update**, **Delete**, Merge	commandes permettant de modifier les données de la base.
DDL : Data Definition Language - Langage de définition des données.	**Create**, **Alter**, **Drop**, Rename, Truncate	commandes permettant de modifier la structure de la base.
TCS : Transaction Control Statement– Langage de contrôle de transaction.	**Commit**, **Rollback**, Savepoint	commandes permettant de gérer les modifications faites par les commandes DML.
DCL : Date Control Language – Langage de contrôle de données.	**Grant**, **Revoke**	commandes permettant de contrôler l'accès des utilisateurs à la base et leurs droits.

2.1 SELECT

```
select nom, salaire, 12*(salaire+300)
from emp
;
```

```
NOM                          SALAIRE 12*(SALAIRE+300)
------------------------- ---------- ----------------
Vabres                          7600            94800
Natter                          5300            67200
Nury                            5200            66000
Quenot                          5500            69600
…
```

Remarque :

- le caractère " ; " permet de terminer la requête.
- l'ordre des colonnes précisées dans la clause SELECT correspond à l'ordre des colonnes affichées dans les résultats.
- à l'affichage les données numériques sont automatiquement justifiées à droite, et les données alpha numériques ou date automatiquement justifiées à gauche.
- tous les titres de colonne sont affichés, par défaut, en majuscule.
- pour sélectionner toutes les colonnes d'une table utiliser le caractère " * ".
- les commandes des requêtes SQL ne sont pas sensibles à la casse.
- les commandes ne peuvent pas être abrégées.
- pour plus de lisibilité, il est conseillé de placer les différentes clauses sur des lignes différentes.
- les opérateurs arithmétiques (+, *, -, /) sont utilisés pour effectuer des opérations sur les nombres et les dates, dans toutes les clauses sauf derrière FROM.
- la présence de parenthèses peut modifier les résultats des opérations.

```
select nom, salaire, 12*salaire*(1+tx_commission/100)
from emp
;
```

```
NOM                          SALAIRE 12*SALAIRE*(1+TX_COMMISSION/100)
------------------------- ---------- --------------------------------
Gallet                          4000                            54000
Schmitt                         4100                            54120
Nguyen                          3890                            53682
Dumas                           3500                            49350
Martinet                        2300
Simon                           2100
Nicolas                         2000
…
```

Une valeur vide est une valeur non renseignée ou inconnue. Elle n'est pas équivalente à zéro ou à un espace. Si une expression arithmétique contient une valeur vide, alors le résultat de l'expression sera vide.

Renommer les colonnes affichées :

```
select nom, nom nom_personne, nom as nom_personne, nom "Nom de la personne"
from emp
;
```

```
NOM             NOM_PERSONNE    NOM_PERSONNE    Nom de la personne
--------------- --------------- --------------- ------------------
Gallet          Gallet          Gallet          Gallet
Schmitt         Schmitt         Schmitt         Schmitt
Nguyen          Nguyen          Nguyen          Nguyen
Dumas           Dumas           Dumas           Dumas
…
```

Utilisation de la clause "distinct" :

```
select distinct nodept from emp
;
```

```
    NODEPT
----------
        10
        31
        32
        33
        34
        35
…
```

```
select nodept from emp
;
```

```
    NODEPT
----------
        10
        31
        31
        32
        33
        34
        34
        35
…
```

L'environnement d'exécution des commandes SQL permet d'utiliser la commande "desc" (abréviation de "describe") qui affiche la structure d'une table.

```
desc emp

Nom                                       NULL ?   Type
----------------------------------------- -------- ---------------
NOEMP                                     NOT NULL NUMBER(7)
NOM                                       NOT NULL VARCHAR2(25)
PRENOM                                             VARCHAR2(25)
USERID                                             VARCHAR2(8)
EMBAUCHE                                           DATE
COMMENTAIRES                                       VARCHAR2(255)
NOSUPR                                             NUMBER(7)
TITRE                                              VARCHAR2(25)
NODEPT                                             NUMBER(7)
SALAIRE                                            NUMBER(11,2)
TX_COMMISSION                                      NUMBER(4,2)
```

Autres commandes utiles :

`@fichier[.ext] :`	Exécuter les commendes SQL contenues dans un fichier (script).
`exit :`	Se déconnecter et fermer la session SQL

2.2 SELECT et restriction, tri

Afficher le nom, le prénom des employés dont le titre est 'Chef Entrepot'.

```
select nom, prenom
from emp
where titre = 'Chef Entrepot'
;
```

```
NOM                       PRENOM
------------------------- --------------
Ubertin                   Charles
Mayer                     Germaine
Boutin                    Gerard
Cazer                     Antoinette
Hazan                     Philippe
```

Afficher le nom, le prénom des employés dont le no de département est 31.

```
select nom, prenom
from emp
where nodept = 31
;
```

```
NOM                       PRENOM
------------------------- ---------
Nury                      Patrick
Mougeot                   Thomas
```

Utilisation d’opérateurs de comparaison

Afficher le nom, le salaire des employés dont le salaire mensuel est strictement inférieur à 2000 euros.

```
select nom, salaire
from emp
where salaire < 2000
;
```

```
NOM                          SALAIRE
------------------------- ----------
Meret                           1800
Perrin                          1750
Delages                         1680
Seron                           1680
```

Afficher le nom, la date d'embauche des employés dont la date d'embauche est strictement inférieure au '01/09/03'.

```
select nom, embauche
from emp
where embauche < '01/09/03'
;
```

```
NOM                       EMBAUCHE
------------------------- --------
Vabres                    26/07/03
Natter                    15/08/03
```

Afficher le nom, le salaire des employés dont le salaire mensuel est compris entre 2000 et 4000 euros en utilisant la clause between.

```
select nom, salaire from emp where salaire between 2000 and 4000 ;
```

équivalent à

```
select nom, salaire from emp where salaire >= 2000 and salaire <= 4000 ;
```

```
NOM                          SALAIRE
------------------------- ----------
Ubertin                         3800
Mayer                           3650
Boutin                          3500
Cazer                           3350
…
```

Afficher le nom des employés dont le nom contient un 'a' en 2ème position en utilisant les caractères joker _ (remplace un caractère) et % (remplace une chaîne).

```
select nom from emp
where nom like '_a%'
;
```

```
NOM
----------------------
Vabres
Natter
Mayer
Cazer
…
```

Afficher les noms des employés dont le titre est contenu dans la liste Secretaire, Representant en utilisant la clause IN.

```
select nom
from emp
where titre in (’Secretaire’, ’Representant’)
;

NOM
------------
Mougeot
Gallet
Schmitt
Nguyen
Dumas
Martinet
…
```

Afficher le nom des employés dont le taux de commission est vide en utilisant IS NULL.

```
select nom from emp where tx_commission is null
;

NOM
----------
Vabres
Natter
Nury
Quenot
Roques
…
```

Pour afficher le nom des employés dont le taux de commission n’est pas vide on utilise IS NOT NULL.

Afficher les noms des employés dont le titre est Secretaire et dont le salaire est strictement supérieur à 2000 euros en utilisant la clause AND.

```
select nom
from emp
where titre = ’Secretaire’
  and salaire > 2000
;

NOM
-----------
Martinet
Simon
Chamart
```

Afficher les noms des employés dont le titre est Secretaire ou dont le salaire est strictement supérieur à 2000 euros en utilisant la clause OR.

```
select nom
from emp
where titre = 'Secretaire'
   or salaire > 2000
;

NOM
-----------
Vabres
Natter
Nury
Quenot
Roques
…
```

Afficher le nom, le salaire des employés dans l'ordre croissant des salaires.

```
select nom, salaire
from emp
order by salaire
;

NOM                          SALAIRE
------------------------- ----------
Delages                        1680
Seron                          1680
Perrin                         1750
Meret                          1800
Nicolas                        2000
…
```

Remarque : le tri peut porter sur une colonne qui n'appartient pas au select

Afficher le nom, le salaire des employés dans l'ordre décroissant des salaires.

```
select nom, salaire
from emp
order by salaire desc
;

NOM                          SALAIRE
------------------------- ----------
Vabres                         7600
Quenot                         5500
Roques                         5500
Natter                         5300
Nury                           5200
…
```

Afficher le nom, la date d'embauche des employés dans l'ordre croissant des dates d'embauche en utilisant dans la clause order by un alias sur la colonne date d'embauche.

```
select nom, embauche "Date emb"
from emp
order by "Date emb"
;
```

```
NOM                       Date emb
------------------------- --------
Vabres                    26/07/03
Natter                    15/08/03
Nury                      04/09/03
Quenot                    24/09/03
…
```

Afficher le nom, le titre, le salaire des employés dans l'ordre croissant des titres et pour un titre donné, dans l'ordre décroissant des salaires.

```
select nom, titre, salaire
from emp
order by titre, salaire desc
;
```

```
NOM                       TITRE                       SALAIRE
------------------------- ------------------------- ----------
Ubertin                   Chef Entrepot                   3800
Mayer                     Chef Entrepot                   3650
Boutin                    Chef Entrepot                   3500
Cazer                     Chef Entrepot                   3350
Hazan                     Chef Entrepot                   3200
Roques                    Dir, Administration             5500
Natter                    Dir, Distribution               5300
...
```

2.3 SELECT et fonction SQL

A partir d'un ou plusieurs arguments en entrée, une fonction SQL réalise un traitement qui fournit en sortie un résultat.

Fonctions de conversion de casse de chaîne de caractères

LOWER(column | expr) : conversion en minuscule.

UPPER(column | expr) : conversion en majuscule.

...

Fonctions de manipulation de chaîne de caractères

LENGTH(column | expr) : retourne le nombre de caractères d'une chaîne

SUBSTR(column | expr, m [,n]) : retourne une chaîne de caractères extraite de la chaîne de caractères column (ou expr) sur une longueur n à partir de la position m.

INSTR(column | expr, c, n) : retourne la position de la première occurrence du caractère c dans la chaîne de caractères column ou expr, à partir de la n ième position.

...

Fonctions de traitement des nombres

ROUND(column | expr [,n]) : retourne l'arrondi de column ou expr à n décimales près, si n est positif l'arrondi est fait après la virgule, si n est négatif l'arrondi est fait avant la virgule (à la dizaine, centaine, etc …).

TRUNC(column | expr [,n]) : retourne la valeur tronquée de column ou de expr à n décimales près, si n est positif la troncature est faite après la virgule, si n est négatif la troncature est faite avant la virgule (à la dizaine, centaine, etc …)..

...

Fonctions de traitement des dates

Opérateurs arithmétiques utilisés avec les dates :

Date + N :	N est exprimé en jour, retourne une date
Date – N :	N est exprimé en jour, retourne une date
Date – Date :	retourne un nombre de jours
Date + N/24 :	N est exprimé en heure, retourne une date

SYSDATE : (pseudo colonne) retourne la date et l'heure courante

MONTHS_BETWEEN (date1,date2) : retourne le nombre de mois séparant deux dates (si date1 est antérieure à la date2, le résultat est négatif, si date1 est plus postérieur à la date2, le résultat est positif.

ADD_MONTHS(date,n) : ajoute ou retranche n mois à une date suivant que n est un entier positif ou négatif.

NEXT_DAY(date, 'nom du jour') : retourne la date du jour ayant ce nom et suivant date.
NEXT_DAY(date, no de jour) : retourne la date du jour ayant ce numéro et suivant date.

LAST_DAY(date) : retourne le dernier jour du mois de date.

ROUND(date [,'format']) : retourne date arrondie à l'unité spécifiée par format. Si le format est omis, date est arrondie au jour le plus près.

TRUNC(date [,'format']) : retourne date tronquée à l'unité spécifiée par format. Si le format est omis, date est tronquée au jour le plus près.
...

Fonctions de conversion :

Remarque : le noyau de la base peut mettre en œuvre des conversions implicites

TO_CHAR(number|date,[fmt],[nlsparams]) :	retourne une chaîne de caractères
TO_NUMBER (char ['format']) :	retourne un nombre
TO_DATE (char [,'format']) :	retourne une date

...

Afficher le nom, la date d'embauche (convertie en chaîne de caractères avec le format Day dd Month yyyy) des employés dans l'ordre croissant de date d'embauche.

```
select nom, to_char(embauche,'Day dd Month yyyy') "Jour d'embauche"
from emp
order by embauche
;
```

```
NOM                       Jour d'embauche
------------------------- -------------------------
Vabres                    Samedi   26 Juillet   2003
Natter                    Vendredi 15 Aout      2003
Nury                      Jeudi    04 Septembre 2003
Quenot                    Mercredi 24 Septembre 2003
…
```

Liste des symboles de masque utilisés pour les conversions de date :

CC	Siècle sur 2 chiffres
YYYY	Année sur 4 chiffres
YY	Année sur les 2 derniers chiffres
YEAR (year)	Année en toutes lettres majuscules (minuscules)
Q	Trimestre en chiffre
MM ou mm	Mois sur 2 chiffres
MONTH (month)	Mois en toutes lettres majuscules (minuscules)
MON (mon)	Mois sur les 3 premières lettres majuscules (minuscules)
RM	Mois en chiffres romains
WW	Semaine de l'année en chiffres
W	Semaine du mois en chiffres
DY (dy)	Jour de la semaine sur les 2 premiers caractères majuscules (minuscules)
DAY (day)	Jour de la semaine en toutes lettres majuscules (minuscules)
FmDAY	Jour de la semaine en toutes lettres majuscules (minuscules) sans les espaces
DDD	Jour de l'année en chiffres
DD	Jour du mois en chiffres
D	Jour de la semaine en chiffres
J	Numéro du jour par rapport à la date 01/01/-4712
HH, HH12, HH24	Heures
AM ou PM	Avant midi ou après midi
MI	Minutes (0,59)
SS	Secondes (0,59)
SSSSS	Secondes (0, 86399)
TH	Nombre ordinal
SP	Nombre en toutes lettres
SPTH ou THSP	Nombre ordinal en toutes lettres

Liste des symboles de masque utilisés pour les conversions de nombre :

9	Autant de 9 que de nombre maximal de chiffres
0	Affichage des zéros au début du nombre
$	Affichage de $ au début
L	Affichage de la devise locale
.	Point décimal
,	Séparateur des milliers
MI	Affichage du signe – à droite du nombre
PR	Affichage d'un nombre négatif entre < et >
E+n ou E-n	Notation scientifique
V	Multiplie autant de fois par 10 qu'il y a de 9 après le V
B	Affiche des espaces à la place des chiffres absents

Si la donnée excède le nombre de positions du masque :

```
select to_char(salaire,'999.99') "Salaire" from emp;

Salaire
--------
#######
#######
```

Les fonctions imbriquées :

Afficher le jour 3 mois après la date d'embauche :

```
select to_char(add_months(embauche,3), 'Day dd-mm-yyyy') "Fin periode d'essai"
from emp;

Fin periode d'essai
--------------------
Dimanche 26-10-2003
Samedi   15-11-2003
Jeudi    04-12-2003
Mercredi 24-12-2003
…
```

La fonction NVL :

La fonction NVL permet de remplacer une valeur vide par une valeur choisie.
Afficher le nom et le taux de commission. Si le taux de commission est vide, afficher 0.

```
select nom, nvl(tx_commission, 0) "Commission" from emp;

NOM                       Commission
------------------------- -----------------
Gallet                               12.5
Schmitt                              10.0
Nguyen                               15.0
Dumas                                17.5
Martinet                              0.0
Simon                                 0.0
…
```

2.4 SELECT et la pseudo-colonne ROWNUM

Pour chaque ligne retournée par une requête, la pseudo-colonne ROWNUM retourne un nombre (1, 2, …) qui indique l'ordre avec lequel Oracle a sélectionné la ligne dans la table.

Afficher les 10 premières lignes sélectionnées dans la table EMP.

```
SELECT *
FROM EMP
WHERE ROWNUM < 11
;
```

Si une clause ORDER BY est utilisée, l'affectation des valeurs à ROWNUM se produit avant la mise en œuvre du tri.
Afficher les 10 premières lignes sélectionnées dans la table EMP. Trier par nom croissant.

```
SELECT * FROM EMP WHERE ROWNUM < 11 ORDER BY nom;
```

La requête suivante retourne aucune ligne.

```
SELECT * FROM EMP WHERE ROWNUM > 1;
```

EX201 : Rechercher le nom, le titre, le no de département, le salaire des employés dont le titre est soit Representant soit Secretaire et qui appartiennent tous au département 34.

EX202 : Rechercher le nom, le titre, le no de département, le salaire des employés dont le titre est Representant ou dont le titre est Secretaire du département 34.

EX203 : Rechercher le nom, la rémunération totale (c'est à dire le salaire augmenté de la commission appliquée au salaire), en tenant compte (fonction NVL) des taux de commission vides pour certains employés.

EX204 : Rechercher le nom et le rang (fonction INSTR) de la lettre 'r' à partir de la 3eme lettre dans le nom, des employés.

EX205 : Rechercher le nom et la date d'embauche des employés embauchés il y a plus d'une année.

EX206 : Afficher le nom, la date d'embauche (renommée et formatée), le premier Vendredi qui suit la fin de la période d'essai (renommée et formatée), le dernier jour du mois de la fin de période d'essai (renommée et formatée).

EX201

```
SELECT NOM, TITRE, NODEPT, SALAIRE
FROM    EMP
WHERE   (TITRE = 'Representant' OR TITRE = 'Secretaire')
  AND   NODEPT = 34
;
```

EX202

```
SELECT NOM, TITRE, NODEPT, SALAIRE
FROM    EMP
WHERE   TITRE = 'Representant'
   OR   (TITRE = 'Secretaire' AND NODEPT = 34)
;
```

EX203

```
SELECT  NOM,
        SALAIRE * (1 + NVL(TX_COMMISSION,0)/100) "Remuneration totale"
FROM    EMP
;
```

EX204

```
SELECT NOM, INSTR(NOM,'r',3)
FROM   EMP
;
```

EX205

```
SELECT  NOM, EMBAUCHE
FROM    EMP
WHERE   MONTHS_BETWEEN(SYSDATE, EMBAUCHE) > 12
;
```

EX206

```
COLUMN  "DATE EMBAUCHE"   FORMAT A15
COLUMN  "FIN ESSAI"       FORMAT A20
COLUMN  "FIN MOIS ESSAI"  FORMAT A15

SELECT  NOM,
  TO_CHAR(EMBAUCHE,'DD-MM-YY')                                    "DATE EMBAUCHE",
  TO_CHAR(NEXT_DAY(add_months(embauche,3),'VENDREDI'),'DAY DD-MM-YY') "FIN ESSAI",
  TO_CHAR(LAST_DAY(add_months(embauche,3)),'DD-MM-YY')            "FIN MOIS ESSAI"
FROM    EMP
;
```

3 SELECT et jointures, sous requêtes, fonctions de groupe.

3.1 Les jointures

Le produit cartésien est l'opération qui permet d'associer toutes les occurrences d'un premier ensemble de cardinalité C1 à toutes les occurrences d'un deuxième ensemble de cardinalité C2. La cardinalité de l'ensemble obtenu est C1 x C2.

(a, b, c) X (1,2) = (a1, a2, b1, b2, c1, c2) 3 x 2 = 6

L'affichage des données appartenant à plusieurs tables se fait par l'opération de jointure. La condition de jointure spécifie les règles qui permettent d'associer les données d'une colonne dans une table avec les données d'une autre colonne dans une autre table.
Ces règles mettent en jeu souvent des colonnes définies comme clé primaire et clé étrangère.

L'équi-jointure

L'équi-jointure permet d'afficher les données provenant de plusieurs tables lorsqu'une valeur dans une colonne d'une table est égale à une valeur d'une autre colonne dans une autre table. Afin d'éviter toute ambiguïté, les noms des colonnes peuvent être qualifiés avec le nom (ou l'alias) de la table à laquelle elles appartiennent.

Afficher le nom, le prénom, le numéro de département des employés et le nom du département correspondant.

```
select e.nom, e.prenom, e.nodept, d.nom
  from emp e, dept d
 where e.nodept = d.nodept
;

NOM             PRENOM              NODEPT NOM
--------------- --------------- ---------- ---------------
Vabres          Pierre                  50 Administration
Natter          Genevieve               41 Distribution
Nury            Patrick                 31 Vente
Quenot          Alain                   10 Finance
…
```

La non équi-jointure

Afficher le numéro du département, le nom, le salaire des employés dont le salaire est supérieur aux salaires des employés du département 31. Afficher (dans la même requête) le nom et le salaire des employés du département 31. Trier sur le salaire.

```
select e.nodept, e.nom, e.salaire, m.nodept, m.nom, m.salaire
  from emp e, emp m
 where e.salaire > m.salaire
   and m.nodept = 31
order by e.salaire desc
;
```

```
    NODEPT NOM                SALAIRE     NODEPT NOM                SALAIRE
---------- -------------- ---------- ---------- -------------- ----------
        50 Vabres                7600         31 Nury                   5200
        50 Vabres                7600         31 Mougeot                3800
        10 Quenot                5500         31 Nury                   5200
        50 Roques                5500         31 Nury                   5200
        10 Quenot                5500         31 Mougeot                3800
        50 Roques                5500         31 Mougeot                3800
        32 Gallet                4000         31 Mougeot                3800
...
```

La jointure externe

La jointure externe (outer join) permet d’afficher les enregistrements qui satisfont le critère de jointure et aussi les enregistrements qui ne satisfont pas le critère de jointure.
L’opérateur de jointure externe (+) est placé dans la condition de jointure du côté de l’égalité où il manque l’information, c'est à dire du côté de la clé étrangère.

Afficher le nom des employés, le nom des départements pour tous les employés et tous les départements.

```
select e.nom, d.nom
  from emp e, dept d
 where e.nodept (+) = d.nodept;
```

```
NOM                       NOM
------------------------- -------
Quenot                    Finance
                          Atelier
                          Atelier
Nury                      Vente
Mougeot                   Vente
...
```

L’auto jointure

L’auto jointure (self join) permet de mettre en œuvre une jointure sur deux colonnes appartenant à la même table.

Afficher le nom et le salaire des employés, le nom et le salaire de leur supérieur direct, pour les employés dont le salaire est supérieur à celui de leur supérieur direct.

```
select e.nom, e.salaire, s.nom, s.salaire from emp e, emp s
where e.nosupr = s.noemp and e.salaire > s.salaire
;
```

```
NOM                         SALAIRE NOM                         SALAIRE
------------------------- ---------- ------------------------- ----------
Chamart                         3400 Cazer                           3350
```

La jointure avec la clause ON

Afficher le nom de l’employé, le nom du département avec la clause on

```
select e.nom, d.nom
  from emp e join dept d
    on e.nodept = d.nodept
;
```

```
NOM                       NOM
------------------------- ---------------
Vabres                    Administration
Natter                    Distribution
Nury                      Vente
Quenot                    Finance
...
```

Une autre écriture équivalente :

```
select e.nom, d.nom
  from emp e inner join dept d
    on e.nodept = d.nodept
;
```

```
NOM                       NOM
------------------------- ---------------
Vabres                    Administration
Natter                    Distribution
Nury                      Vente
Quenot                    Finance
...
```

Afficher le nom des employés, le nom des départements pour tous les employés et tous les départements avec une jointure externe droite et ensuite une jointure externe gauche.

```
select e.nom, d.nom
  from emp e right outer join dept d
    on e.nodept = d.nodept
;

NOM                          NOM
---------------------------- --------------------------
Quenot                       Finance
                             Atelier
                             Atelier
Nury                         Vente
Mougeot                      Vente
...
```

ou bien

```
select e.nom, d.nom
  from dept d left outer join emp e
    on e.nodept = d.nodept
;

NOM                          NOM
---------------------------- --------------------------
Quenot                       Finance
                             Atelier
                             Atelier
Nury                         Vente
Mougeot                      Vente
...
```

EX207 : Rechercher le prénom des employés et le nom de la région de leur département.

EX208 : Rechercher le nom du client, le numéro des commandes, le numéro des produits achetés par le client IBM. Proposer deux types d'écriture.

EX209 : Rechercher le no, le nom et le prix des produits qui n'ont pas encore fait l'objet d'une commande. Proposer trois types d'écriture.

EX210 : Rechercher le nom de l'employé Dumas et le nom de son département, omettre volontairement le critère de jointure.

EX207

```
SELECT  PRENOM, REGION.NOM
FROM    EMP,DEPT,REGION
WHERE   EMP.NODEPT = DEPT.NODEPT
  AND   DEPT.NOREGION = REGION.NOREGION
;
```

EX208

```
SELECT CL.NOM, CM.NOCOM, LG.NOPRODUIT
  FROM CLIENT CL, COM CM, LGNCOM LG
 WHERE CL.NOCLIENT = CM.NOCLIENT
   AND CM.NOCOM = LG.NOCOM
   AND CL.NOM = 'IBM'
;

SELECT CL.NOM, CM.NOCOM, LG.NOPRODUIT
  FROM CLIENT CL JOIN COM CM    ON CL.NOCLIENT = CM.NOCLIENT
                 JOIN LGNCOM LG ON    CM.NOCOM = LG.NOCOM
 WHERE CL.NOM = 'IBM'
;
```

EX209

```
SELECT PRODUIT.NOPRODUIT, NOM, PRIX_CATALOGUE
  FROM LGNCOM, PRODUIT
 WHERE LGNCOM.NOPRODUIT (+) = PRODUIT.NOPRODUIT
   AND LGNCOM.NOPRODUIT IS NULL
;

SELECT PRODUIT.NOPRODUIT, NOM, PRIX_CATALOGUE
  FROM LGNCOM RIGHT OUTER JOIN PRODUIT
    ON LGNCOM.NOPRODUIT = PRODUIT.NOPRODUIT
 WHERE LGNCOM.NOPRODUIT IS NULL
;

SELECT PRODUIT.NOPRODUIT, NOM, PRIX_CATALOGUE
  FROM PRODUIT LEFT OUTER JOIN LGNCOM
    ON LGNCOM.NOPRODUIT = PRODUIT.NOPRODUIT
 WHERE LGNCOM.NOPRODUIT IS NULL
;
```

EX210

```
SELECT  EMP.NOM, DEPT.NOM
FROM    EMP,DEPT
WHERE   EMP.NOM = 'Dumas'
;
```

3.2 Les sous requêtes

Les sous requêtes derrière WHERE

Afficher le nom et le salaire des employés dont le salaire est supérieur au salaire de Dumas.

```
select nom, salaire
  from emp
 where salaire > (select salaire
                   from emp
                  where nom = 'Dumas')
;
```

```
NOM                          SALAIRE
------------------------- ----------
Vabres                          7600
Natter                          5300
Nury                            5200
Quenot                          5500
Roques                          5500
…
```

Remarque : la sous requête ramène une seule valeur.

Si la sous requête ramène plusieurs valeurs, il faut utiliser l'opérateur ALL ou l'opérateur ANY

Utilisation de l’opérateur ALL :

Opérateur	Correspond à
< ALL	plus petit que le minimum.
> ALL	plus grand que le maximum.

Afficher le nom et le salaire des employés dont le salaire est supérieur à tous les salaires du département 31.

```
select nom, salaire from emp
 where salaire > all (select salaire from emp
                       where nodept = 31)
;
```

```
NOM                          SALAIRE
------------------------- ----------
Vabres                          7600
Natter                          5300
Quenot                          5500
Roques                          5500
```

Utilisation de l'opérateur ANY :

Opérateur	Correspond à
< ANY	plus petit que le maximum.
> ANY	plus grand que le minimum.
= ANY	à IN.

Afficher le nom et le salaire des employés dont le salaire est supérieur à au moins un salaire du département 31.

```
select nom, salaire from emp
 where salaire > any (select salaire from emp
                      where nodept = 31)
;

NOM                          SALAIRE
------------------------- ----------
Vabres                          7600
Natter                          5300
Nury                            5200
Quenot                          5500
...
```

Les sous requêtes derrière FROM

Remarque : le résultat de la sous requête est "utilisé" comme un tableau temporaire de résultats.

Afficher les 5 plus hauts salaires de la table EMP. Utilisation de ROWNUM avec une sous requête derrière FROM.

```
SELECT *
FROM (SELECT * FROM EMP ORDER BY salaire desc)
WHERE ROWNUM < 6
;
```

Les sous requêtes derrière SELECT

Afficher le nom, le salaire et l'écart entre le salaire et le salaire du President. Trier par salaire décroissant.

```
select nom, salaire,
       salaire - (select salaire from emp where titre = 'President') "Ecart"
  from emp
 order by salaire desc
;
```

```
NOM                        SALAIRE      Ecart
-------------------------- ---------- ----------
Vabres                           7600          0
Quenot                           5500      -2100
Roques                           5500      -2100
Natter                           5300      -2300
Nury                             5200      -2400
...
```

EX211 : Rechercher le nom et le titre des employés qui ont le même titre que Dumas.

EX212 : Rechercher le nom, le salaire et le no de département des employés qui gagnent plus qu'au moins un employé du département 31, classés par no de département et salaire.

EX213 : Rechercher le nom, le salaire et le no de département des employés qui gagnent plus que tous les employés du service 31, classés par no de département et salaire.

EX214 : Rechercher le nom et le titre des employés du service 31 qui ont un titre que l'on ne trouve pas dans le service 32.

EX211

```
SELECT  NOM, TITRE
  FROM    EMP
 WHERE   TITRE = (SELECT   TITRE
                    FROM     EMP
                   WHERE    NOM = 'Dumas')
;
```

EX212

```
SELECT   NOM, SALAIRE, NODEPT
  FROM     EMP
 WHERE    SALAIRE > ANY (SELECT SALAIRE
                           FROM EMP
                          WHERE NODEPT = 31)
ORDER BY 3, 2
;
```

EX213

```
SELECT   NOM, SALAIRE, NODEPT
  FROM     EMP
 WHERE    SALAIRE > ALL (SELECT SALAIRE
                           FROM EMP
                          WHERE NODEPT = 31)
ORDER BY 3, 2
;
```

EX214

```
SELECT  NOM, TITRE
  FROM    EMP
 WHERE   NODEPT = 31
   AND   TITRE NOT IN (SELECT TITRE
                         FROM EMP
                        WHERE NODEPT = 32)
;
```

3.3 Les fonctions de groupe

Elles sont utilisées pour afficher des données relatives à des groupes d'enregistrements.

Quelques fonctions de groupe

Fonction	Commentaire
SUM (expr)	la somme de toutes les valeurs du groupe
MIN (expr)	la plus petite valeur du groupe
MAX (expr)	la plus grande valeur du groupe
COUNT (expr)	le nombre d'enregistrements contenus dans le groupe
AVG (expr)	la moyenne des valeurs du groupe
STDDEV (expr)	l'écart type des valeurs du groupe
VARIANCE (expr)	la variance des valeurs du groupe
...	…

Afficher la somme, le minimum, le maximum, le dénombrement, la moyenne, l'écart type, la variance des salaires des employés.

```
select sum(salaire), min(salaire), max(salaire), count(salaire), avg(salaire),
       stddev(salaire), variance(salaire)
from emp;
```

```
SUM(SALAIRE) MIN(SALAIRE) MAX(SALAIRE) COUNT(SALAIRE) AVG(SALAIRE) STDDEV(SALAIRE) VARIANCE(SALAIRE)
------------ ------------ ------------ -------------- ------------ -------------- -----------------
       86600         1680         7600             25         3464      1509,99448        2280083,33
```

Afficher le nombre de titres distincts des employés.

```
select count(distinct titre) from emp;
```

```
COUNT(DISTINCTTITRE)
--------------------
                   8
```

Afficher la moyenne des taux de commission non vides et la moyenne de tous les taux de commission.

```
select avg(tx_commission), avg(nvl(tx_commission, 0)) from emp;
```

```
AVG(TX_COMMISSION) AVG(NVL(TX_COMMISSION,0))
------------------ --------------------------
                13                        2,6
```

La clause GROUP BY

La clause group by permet d'exprimer la manière dont les enregistrements sont groupés.

Remarque : les colonnes du select qui n'interviennent pas dans une fonction de groupe doivent apparaître dans la clause group by.

Afficher le no de département et la moyenne des salaires des employés par département.

```
select nodept, avg(salaire)
  from emp
 group by nodept
;
```

```
    NODEPT AVG(SALAIRE)
---------- ------------
        10         5500
        31         4500
        32         4000
        33         4100
…
```

Afficher le no de département, le titre et la moyenne des salaires des employés par département et titre.

```
select nodept, titre, avg(salaire)
  from emp
 group by nodept, titre
;
```

```
    NODEPT TITRE                     AVG(SALAIRE)
---------- ------------------------- ------------
        10 Dir, Finance                      5500
        31 Dir, Vente                        5200
        31 Representant                      3800
...
```

La restriction au niveau d'un groupe est obtenue par l'utilisation de la clause HAVING.

Afficher le titre et la moyenne des salaires par titre, pour les moyennes supérieures à 5000. Trier par moyenne de salaire décroissante.

```
select titre, avg(salaire)
  from emp
 group by titre
having avg(salaire) > 5000
 order by avg(salaire) desc
;
```

```
TITRE                     AVG(SALAIRE)
------------------------- ------------
President                         7600
Dir, Administration               5500
Dir, Finance                      5500
Dir, Distribution                 5300
Dir, Vente                        5200
```

EX215 : Rechercher le nom et le salaire de l'employé dont le salaire est le plus grand.

EX216 : Rechercher le no de département, le titre et le nombre d'employés groupés par département, titre.

EX217 : Rechercher les titres et le nombre d'employés pour les titres représentés plus de 2 fois.

EX218 : Rechercher les titres et la moyenne des salaires par titre dont la moyenne est supérieure à la moyenne des salaires des Representants.

EX219 : Rechercher le nombre de salaires renseignés et le nombre de taux de commission renseignés.

EX220 : Rechercher le no de département, la moyenne des salaires de ce département, le nom et le salaire des employés dont le salaire est inférieur à la moyenne des salaires de leur département.

EX215

```
SELECT  NOM, SALAIRE
  FROM    EMP
 WHERE   SALAIRE = (SELECT MAX(SALAIRE) FROM EMP)
;
```

EX216

```
SELECT NODEPT, TITRE, COUNT(*)
  FROM EMP
GROUP BY NODEPT, TITRE
;
```

EX217

```
SELECT TITRE, COUNT(*)
  FROM EMP
 GROUP BY TITRE
HAVING COUNT(*) > 2
;
```

EX218

```
SELECT TITRE, AVG(SALAIRE)
  FROM EMP
 GROUP BY TITRE
HAVING AVG(SALAIRE) > (SELECT AVG(SALAIRE)
                         FROM EMP
                        WHERE TITRE = 'Representant')
;
```

EX219 : Rechercher le nombre de salaires renseignés et le nombre de taux de commission renseignés.

```
SELECT COUNT(SALAIRE),COUNT(TX_COMMISSION) FROM EMP
;
```

EX220

```
SELECT M.NODEPT, M.MOYENNE, E.NOM, E.SALAIRE
  FROM EMP E, (SELECT NODEPT, AVG(SALAIRE) MOYENNE FROM EMP
               GROUP BY NODEPT) M
 WHERE E.NODEPT = M.NODEPT
   AND E.SALAIRE < M.MOYENNE
;
```

4 Les commandes LMD et LDD

4.1 La commande INSERT

Cette commande permet d'insérer un enregistrement dans une table.
Remarque : cette commande met en œuvre une transaction qui doit se terminer par COMMIT (explicite ou implicite) ou par ROLLBACK.

Insérer dans la table dept deux enregistrements.

```
desc dept
 Nom                                       NULL ?   Type
 ----------------------------------------- -------- ------------
 NODEPT                                    NOT NULL NUMBER(7)
 NOM                                       NOT NULL VARCHAR2(25)
 NOREGION                                           NUMBER(7)

insert into dept (nodept, noregion, nom) values (55, 1, 'Marketing')
;
1 ligne créée.

insert into dept values (56, 'Fabrication', 1)
;
1 ligne créée.

commit;
```

Remarque : l'insertion d'un enregistrement avec des valeurs vides se fait soit, en omettant la colonne derrière clause `INSERT INTO nom_de_table` ou en exprimant la valeur par NULL.

Insérer dans la table emp un enregistrement en ne tenant compte que de noemp, nom, embauche et nodept.

```
insert into emp (noemp, nom, embauche, nodept)
values (30, 'Lazard', to_date('14-12-2005','dd-mm-yyyy'), 10)
;
1 ligne créée.

commit;
```

Insérer dans la table bonus, les lignes de la table emp dont le titre est Chef Entrepot ou dont le taux de commission est supérieur à 14.

```
insert into bonus
select      nom, titre, salaire, tx_commission
  from      emp
 where      titre = 'Chef Entrepot'
    or      tx_commission > 14
;

7 ligne(s) créée(s).

commit;
```

4.2 La commande UPDATE

Cette commande permet de modifier les valeurs des colonnes d'une table.
Remarque : cette commande met en œuvre une transaction qui doit se terminer par COMMIT (explicite ou implicite) ou par ROLLBACK.

Augmenter le salaire de Dumas de 5 %.

```
update emp
set salaire = salaire * 1.05
  where nom = 'Dumas'
;

1 ligne mise à jour.

commit;
```

Modifier le salaire et le nodept de Simon en utilisant le salaire et le nodept de Dumas.

```
update emp
set (salaire, nodept) = (select salaire, nodept
                           from emp
                          where nom = 'Dumas')
where nom = 'Simon'
;

1 ligne mise à jour.

commit;
```

Augmenter de 5% le salaire des employés de emp qui sont présents dans la table bonus.

```
update emp
set salaire = salaire * 1.05
where nom in (select nom from bonus)
;

5 ligne(s) mise(s) à jour.

commit;
```

4.3 La commande DELETE

Cette commande permet de supprimer un ou plusieurs enregistrements d'une table.
Remarque : cette commande met en œuvre une transaction qui doit se terminer par COMMIT (explicite ou implicite) ou par ROLLBACK.

```
DELETE FROM BONUS
WHERE       TITRE = 'Chef Entrepot'
;

5 ligne(s) supprimée(s).

commit;
```

Remarque sur les commandes INSERT, UPDATE, DELETE.

Gestion de l'environnement d'exécution de SQL :

SET AUTOCOMMIT ON : la mise à jour effective de la base sera automatique
(COMMIT automatique ou implicite)

SET AUTOCOMMIT OFF : la mise à jour effective de la base sera faite seulement après la commande COMMIT
(COMMIT non automatique ou explicite)

La commande ROLLBACK (ne pas prendre en compte les modifications n'ayant pas fait l'objet de COMMIT) n'est effective que dans le cas de AUTOCOMMIT OFF et avant le COMMIT.

Si sortie normale de l'environnement d'exécution de SQL par EXIT alors COMMIT implicite,
si sortie anormale (panne ...) alors ROLLBACK automatique à la prochaine session.

Par défaut AUTOCOMMIT OFF.

EX221 : Insérer une nouvelle ligne dans la table EMP avec le no d'employé non renseigné.

EX222 : Insérer une nouvelle ligne dans la table EMP (embauche = date système).

EX223 : Augmenter de 5% les employés de la table EMP qui se trouvent dans la table BONUS.

EX224 : Essayer de supprimer l'employé Dumas de la table EMP. Expliquer la situation. Supprimer un autre employé. Vérifier par Select, faire ROLLBACK, vérifier par Select.

EX221

```
INSERT INTO EMP
VALUES (null,'Nogues','Annie','nogann','11/09/03',null,1,
       'Secretaire',50,1830,null)
;
```

EX222

```
INSERT INTO EMP (NOEMP, NOM, EMBAUCHE, NODEPT)
VALUES (1004,'Felix',SYSDATE,50)
;
```

EX223

```
UPDATE  EMP
   SET  SALAIRE = SALAIRE * 1.05
WHERE   NOM IN (SELECT NOM FROM BONUS)
;
```

EX224

```
DELETE  FROM EMP
WHERE   NOM = 'Dumas'
;
```

Dumas est un employé dont le noemp (clé primaire) est utilisé dans novendeur (clé étrangère) de la table CLIENT.

```
DELETE  FROM EMP
WHERE   NOM = 'Seron'
;

SELECT * FROM EMP WHERE NOM = 'Seron'
;

ROLLBACK
;

SELECT * FROM EMP WHERE NOM = 'Seron'
;
```

4.4 La commande CREATE TABLE (syntaxe minimum)

Créer la table client :

```
CREATE TABLE client
(noclient                   NUMBER(7),
 nom                        VARCHAR2(50) CONSTRAINT client_nom_nn NOT NULL,
 telephone                  VARCHAR2(25),
 adresse                    VARCHAR2(400),
 ville                      VARCHAR2(30),
 etat                       VARCHAR2(20),
 pays                       VARCHAR2(30),
 departement                VARCHAR2(75),
 niveau_credit              VARCHAR2(1),
 novendeur                  NUMBER(7),
 noregion                   NUMBER(7),
 commentaires               VARCHAR2(255),
     CONSTRAINT client_noclient_pk PRIMARY KEY (noclient),
     CONSTRAINT client_niveau_credit_ck CHECK (niveau_credit IN ('E', 'B', 'M')))
;
```

Quelques exemples de types de données :

TYPE	**DESCRIPTION**	**Minimum**	**Maximum**	**Exemples de valeurs**
VARCHAR2 (taille)	Chaîne de caractères de longueur variable	1 car.	4000 car.	'A'
NUMBER [(taille[,precision])]	Numérique. (prec<=38, exposant max -84 +127)	10 exp -84	10 exp 127	10.9999
LONG (taille)	Chaîne de caractères de longueur variable.	1 car.	4 giga car.	'AAAHHHHHH...HHH'
DATE	Date (du siècle à la seconde)	01/01/-4712 (avant J.C)	31/12/9999	'10-FEB-04' '10/02/04' (dépend du format d'affichage ou du paramétrage local)
RAW (taille)	Données binaires devant être entrées en notation hexadécimale. Taille : 1 à 255 caractères	1 octet	4000 octets	
LONG RAW (taille)	Données binaires devant être entrées en notation hexadécimale.	1 octet	4 GO	
CHAR (taille)	Chaîne de caractères de longueur fixe	1 car.	4000 car.	'AZERTY'
……..	……..	……	……	……..

CREATE TABLE peut être utilisé avec une sous requête :

```
CREATE TABLE emp30 (noemp, nom, salaire, nodept)
          AS select noemp, nom, salaire, nodept
               from emp
              where nodept > 30
;
```

4.5 La commande ALTER TABLE

```
ALTER TABLE nom_table ADD (column datatype [DEFAULT expr] [NOT NULL]
                          [,column datatype]…)
;

ALTER TABLE nom_table MODIFY (column datatype [DEFAULT expr]
                             [,column datatype] …);


ALTER TABLE nom_table DROP COLUMN nom_colonne
;
```

4.6 La commande DROP TABLE :

```
DROP TABLE nom_table
;
```

4.7 Les contraintes

Type	
NOT NULL	Permet de contrôler que la colonne ne peut pas contenir de valeur vide
UNIQUE	Permet de contrôler que la ou les colonne(s) contient (contiennent) une valeur unique pour tous les enregistrements de la table
PRIMARY KEY	Permet de créer la clé primaire de la table
FOREIGN KEY	Permet de créer une clé étrangère en relation (REFERENCES) avec une clé primaire d'une autre table ON DELETE CASCADE : la suppression d'un enregistrement de la table maître entraîne la suppression des enregistrements de la table détail ON DELETE SET NULL : la suppression d'un enregistrement de la table maître entraîne la mise à jour avec NULL de la colonne de la table détail
CHECK	Permet de contrôler que la colonne satisfait les conditions définies.

Ajouter une contrainte :

```
ALTER TABLE emp
ADD CONSTRAINT emp_nodept_fk
FOREIGN KEY (nodept) REFERENCES dept (nodept)
;
```

Supprimer une contrainte :

```
ALTER TABLE dept
DROP CONSTRAINT dept_nodept_pk CASCADE;
```

Remarque : `CASCADE` supprime les contraintes de clés étrangères qui se référaient à l'ancienne clé primaire

Utilisation de CASCADE CONSTRAINTS

```
DROP TABLE nom_table CASCADE CONSTRAINTS
;
```

4.8 La commande CREATE VIEW (syntaxe minimum)

Une vue est une représentation logique des données appartenant à une ou plusieurs tables.
Une vue permet de restreindre l'accès aux données et permet de simplifier l'écriture des requêtes.

Créer un vue permettant d'accéder au noemp, nom, salaire, titre des employés dont le salaire est supérieur à 3200.

```
CREATE VIEW vemp3200
AS select noemp, nom, salaire, titre
     from emp
    where salaire > 3200
;
```

4.9 La commande DROP VIEW :

```
DROP VIEW vemp3200;
```

Une vue, basée directement sur une table, peut servir à mettre à jour les données de la table.

Remarque : utilisation du dictionnaire des données

Par exemple,

```
desc user_tables
select table_name from user_tables;

desc user_views
select view_name from user_views;

desc user_constraints
select constraint_name from user_constraints;
```

4.10 Le script de création des tables utilisées pour les exercices.

5 SQL, compléments

Modifier son mot de passe :

```
alter user nom_utilisateur identified by nouveau_mot_de_passe;
```

L’expression CASE :

L’expression CASE permet d’utiliser la structure de IF-THEN-ELSE sans recourir à l’utilisation de PL/SQL.

Afficher le nom, le salaire et le niveau de salaire des employés.

```
select nom, salaire, case when salaire < 2000 then 'Niveau 1'
                          when salaire < 3000 then 'Niveau 2'
                          when salaire < 4000 then 'Niveau 3'
                          else 'Niveau 4'
                     end "Niveau de salaire"
from emp;

NOM                          SALAIRE Niveau de salaire
-------------------------- ---------- -----------------
Gallet                           4000 Niveau 4
Schmitt                          4100 Niveau 4
Nguyen                           3890 Niveau 3
Dumas                            3500 Niveau 3
Martinet                         2300 Niveau 2
Simon                            2100 Niveau 2
```

Les opérateurs ensemblistes

INTERSECT :	permet de ramener les enregistrements communs aux résultats retournés par tous les select
UNION	permet de ramener les enregistrements appartenant aux résultats retournés par l'un ou l'autre des select (ne ramène qu'une seule fois les lignes communes)
MINUS	permet de ramener les enregistrements appartenant au résultat retourné par le premier select qui n'appartiennent pas au résultat retourné par le second select

...

Les enregistrements manipulés doivent être de même structure (même nombre de colonnes et même type de colonne).

```
select  nom, 'Salaire   ' "Type", salaire  "Montant"
from emp
union
select  nom, 'Commission' " Type ", salaire * tx_commission/100 " Montant "
from emp where tx_commission is not null
;

select nodept "Dep." from emp
intersect
select nodept "Dep." from dept
;

select nodept "Dep." from dept
minus
select nodept "Dep." from emp
;
```

Les sous requêtes synchronisées

Une sous requête synchronisée est une sous requête imbriquée qui est exécutée pour chaque ligne traitée par la requête principale, et qui utilise les valeurs d'une colonne retournée par la requête principale. La synchronisation est obtenue en utilisant dans la sous requête, un élément de la requête principale.

Afficher le nom et le salaire des employés qui ont un salaire inférieur à la moyenne de leur département.

```
select nom, salaire from emp e
where salaire < (select avg(salaire) from emp
                  where nodept = e.nodept)
;

NOM                          SALAIRE
------------------------- ----------
Roques                          5500
Cazer                           3350
Mougeot                         3800
Martinet                        2300
Simon                           2100
…
```

Afficher le nom et le titre des employés qui sont les supérieurs d'autres employés.

```
select nom, titre from emp e
where exists (select null from emp
               where emp.nosupr = e.noemp)
;

NOM                       TITRE
------------------------- --------------------
Vabres                    President
Natter                    Dir, Distribution
Nury                      Dir, Vente
Ubertin                   Chef Entrepot
Mayer                     Chef Entrepot
…
```

Afficher le nom et le no des départements auxquels personne n'est affecté.

```
select nom, nodept
from dept d
where not exists (select null from emp e
                   where e.nodept = d.nodept)
;

NOM                          NODEPT
------------------------- ----------
Atelier                          20
Atelier                          30
```

EX225 : Créer une vue (nom : DEPT4) relative au no, nom et titre des employés des départements supérieurs à 40 et utiliser la vue pour interroger. Mettre à jour EMP et utiliser la vue pour interroger.

EX226 : Mettre à jour la table EMP en utilisant la vue DEPT4.

EX227 : Rechercher le prénom si le département est 41 sinon le nom des employés, le no de département, et le salaire des employés classé par no de département.

EX228 : Afficher la liste des vues du dictionnaire, et accéder aux informations de certaines vues.

EX225

```
CREATE VIEW DEPT4 AS
SELECT NOEMP, NOM, TITRE FROM EMP  WHERE NODEPT > 40
;
SELECT * FROM DEPT4
;
SELECT NOM, TITRE FROM DEPT4
 WHERE  NOEMP > 20
;
UPDATE  EMP
   SET  TITRE = 'Chef Entrepot'
 WHERE  NOM = 'Chamart'
;
SELECT * FROM DEPT4
;
COMMIT
;
```

EX226

```
UPDATE DEPT4
   SET TITRE = 'Chef Entrepot' WHERE NOM = 'Seron'
;
SELECT * FROM DEPT4
;
COMMIT
;
```

EX227

```
SELECT CASE when nodept = 41 then prenom else nom END "NOM OU PRENOM",
       NODEPT, SALAIRE FROM EMP ORDER BY 2
;
```

EX228

```
SELECT * FROM DICT
;
```

MODELE PHYSIQUE des DONNEES