

CHAPITRE 5.8:
**CRÉER ET GÉRER DES
TABLES**

Dr Coulibaly Tiekoura

PLAN DU CHAPITRE

1. Extraire des données à l'aide de l'instruction SQL SELECT
2. Restreindre et trier les données
3. Utiliser des fonctions monolignes afin de personnaliser la sortie
4. Afficher des données agrégées à l'aide des fonctions de groupe
5. Afficher des données de plusieurs tables
6. Utiliser des sous -interrogations
7. Utiliser des opérateurs ensemblistes

PLAN DU CHAPITRE (SUITE)

- 8. Manipuler des données
- 9. Utiliser des instructions LDD pour créer et gérer des tables

OBJECTIFS

- ◉ Répartir les principaux objets de base de données en catégories.
- ◉ Modifier la structure d'une table
- ◉ Répertorier les types de données disponibles pour les colonnes
- ◉ Créer une table simple
- ◉ Comprendre la façon dont les contraintes sont créées lors de la création de la table.
- ◉ Décrire le fonctionnement des objets.

OBJETS DE BASE DE DONNÉES

Object	Description
Table	Unité de stockage de base; constituée de lignes
View	Représente de façon logique des sous-ensembles de données d'une ou plusieurs tables
Séquence	Génère des valeurs numériques
Index	Améliore les performances de certaines interrogations
Synonyme	Affecte d'autres noms aux objets

RÈGLES D'APPELLATION

⦿ Les noms des tables et des colonnes:

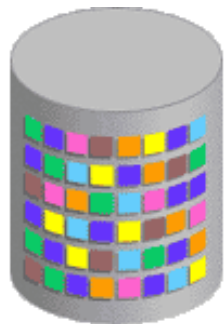
- Doivent commencer par une lettre
- Doivent comporter de 1 à 30 caractères
- Doivent contenir uniquement les caractères A-Z, a-z, 0-9, _, \$ et #
- Ne doivent pas dupliquer le nom d'un autre objet appartenant au même utilisateur
- Ne doivent pas être un mot réservé du serveur Oracle.

INSTRUCTION CREATE TABLE

```
CREATE TABLE [schema.]table  
    (column datatype [DEFAULT expr][, ...]);
```

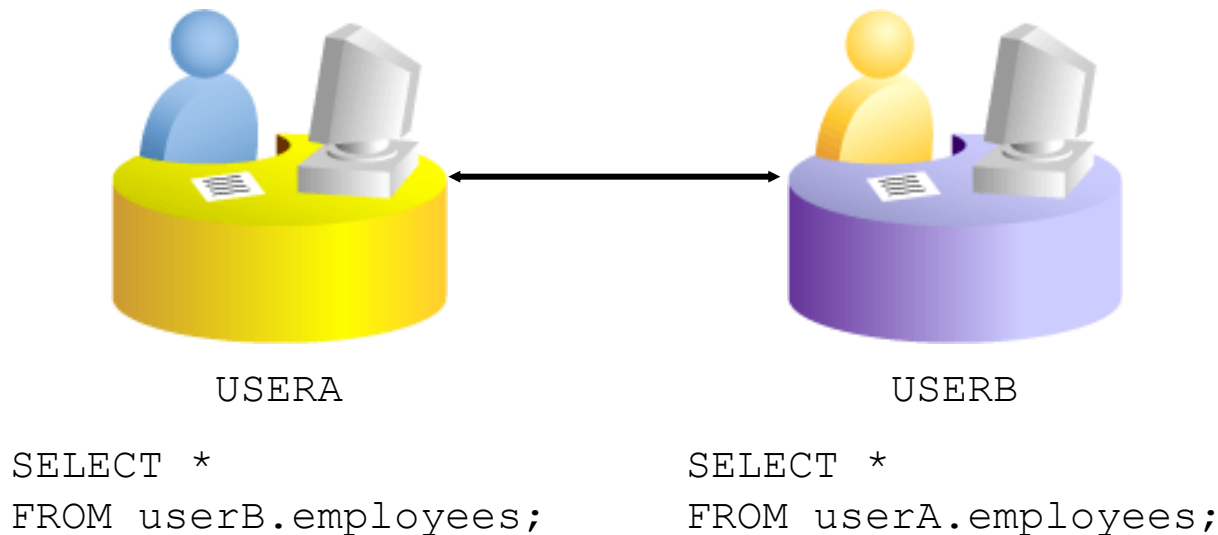
● Indiquez:

- Le nom de la table
- Le nom de la colonne, son type de données et sa taille.



RÉFÉRENCER LES TABLES D'UN AUTRE UTILISATEUR

- Les tables appartenant à d'autres utilisateurs ne résident pas dans le schéma de l'utilisateur.
- Vous devez utiliser le nom du propriétaire comme préfixe pour ces tables.



OPTION DEFAULT

- Indiquez une valeur par défaut pour une colonne lors de l'insertion.

```
... hire_date DATE DEFAULT SYSDATE, ...
```

- Le type de données par défaut doit correspondre au type de colonne.

```
CREATE TABLE hire_dates  
    (id          NUMBER(8),  
     hire_date DATE DEFAULT SYSDATE);  
Table created.
```

CRÉER DES TABLES

● Créez la table

```
CREATE TABLE dept  
    (deptno      NUMBER(2),  
     dname       VARCHAR2(14),  
     loc         VARCHAR2(13),  
     create_date DATE DEFAULT SYSDATE);
```

Table created.

● Vérifiez la création de la table

```
DESCRIBE dept
```

Name	Null?	Type
DEPTNO		NUMBER(2)
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)
CREATE_DATE		DATE

TYPES DE DONNÉES

Data Type	Description
VARCHAR2 (<i>size</i>)	Données de type caractère de longueur variable
CHAR (<i>size</i>)	Données de type caractère de longueur fixe
NUMBER (<i>p</i> , <i>s</i>)	Données numériques de longueur variable
DATE	Valeur de date et d'heure
LONG	Données de type caractère de longueur variable (jusqu'à 2 GB)

INCLURE DES CONTRAINTES

- ⦿ Les contraintes appliquent des règles au niveau table.
- ⦿ Les contraintes empêchent la suppression d'une table s'il existe des dépendances.
- ⦿ Les types de contrainte suivants sont pris en charge:
 - NOT NULL
 - UNIQUE
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK

RÈGLES RELATIVES AUX CONTRAINTES

- ⦿ Vous pouvez créer une contrainte à différents instants:
 - Lors de la création de la table
 - Après la création de la table
- ⦿ Définissez une contrainte au niveau colonne ou au niveau table.

DÉFINIR DES CONTRAINTES

⦿ Syntaxe

```
CREATE TABLE [schema.] table  
    (column datatype [DEFAULT expr]  
    [column_constraint],  
    ...  
    [table_constraint] [, ...]);
```

⦿ Contrainte au niveau colonne

```
column [CONSTRAINT constraint_name] constraint_type,
```

⦿ Contrainte au niveau table

```
column, ...  
[CONSTRAINT constraint_name] constraint_type  
(column, ...),
```

DÉFINIR DES CONTRAINTES

⦿ Contrainte au niveau colonne:

```
CREATE TABLE employees(  
  employee_id  NUMBER(6)  
    CONSTRAINT emp_emp_id_pk PRIMARY KEY,  
  first_name   VARCHAR2(20),  
  ...);
```

1

⦿ Contrainte au niveau table

```
CREATE TABLE employees(  
  employee_id  NUMBER(6),  
  first_name   VARCHAR2(20),  
  ...  
  job_id       VARCHAR2(10) NOT NULL,  
  CONSTRAINT emp_emp_id_pk  
    PRIMARY KEY (EMPLOYEE_ID));
```

2

CONTRAINTE NOT NULL

- Garantit que les valeurs NULL ne sont pas autorisées pour la colonne :

EMPLOYEE_ID	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	DEPARTMENT_ID
100	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000	90
101	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000	90
102	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000	90
103	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000	60
104	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	6000	60
178	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA_REP	7000	
200	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	4400	10

...

20 rows selected.



Contrainte NOT NULL (aucune ligne ne peut contenir de valeur NULL pour cette colonne)



Contrainte NOT NULL



Absence de NOT NULL contrainte (n'importe quelle ligne peut contenir une valeur NULL pour cette colonne)

CONTRAINTE UNIQUE

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	EMAIL
100	King	SKING
101	Kochhar	NKOCHHAR
102	De Haan	LDEHAAN
103	Hunold	AHUNOLD
104	Ernst	BERNST

...



INSERT INTO

208	Smith	JSMITH
209	Smith	JSMITH

← Autorisé

← Non autorisé :
existe déjà

Contrainte UNIQUE

CONTRAİNTE UNIQUE


- Définie au niveau table ou au niveau colonne

```
CREATE TABLE employees (  
    employee_id      NUMBER(6),  
    last_name        VARCHAR2(25) NOT NULL,  
    email            VARCHAR2(25),  
    salary           NUMBER(8,2),  
    commission_pct   NUMBER(2,2),  
    hire_date        DATE NOT NULL,  
    ...  
    CONSTRAINT emp_email_uk UNIQUE(email));
```

CONTRAINTE PRIMARY KEY

DEPARTMENTS

PRIMARY KEY



DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500

...

Non autorisé
(valeur null)



INSERT INTO



	Public Accounting		1400
50	Finance	124	1500

Non autorisé
(50 existe déjà)

CONTRAINTE FOREIGN KEY

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500

PRIMARY
KEY



...

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
107	Lorentz	60

FOREIGN
KEY



INSERT INTO

200	Ford	9
201	Ford	60

Non autorisé
(9 n'existe pas)



Autorisé



CONTRAİNTE FOREIGN KEY

- Définie au niveau table ou au niveau colonne :

```
CREATE TABLE employees(  
    employee_id      NUMBER(6),  
    last_name        VARCHAR2(25) NOT NULL,  
    email            VARCHAR2(25),  
    salary           NUMBER(8,2),  
    commission_pct   NUMBER(2,2),  
    hire_date        DATE NOT NULL,  
    ...  
    department_id    NUMBER(4),  
    CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)  
        REFERENCES departments(department_id),  
    CONSTRAINT emp_email_uk UNIQUE(email));
```

CONTRAÎNTE FOREIGN KEY : MOTS CLÉS

- ◉ FOREIGN KEY : définit la colonne dans la table enfant au niveau contrainte de table.
- ◉ REFERENCES : identifie la table et la colonne dans la table parent.
- ◉ ON DELETE CASCADE : supprime les lignes dépendantes dans la table enfant lorsqu'une ligne de la table parent est supprimée.
- ◉ ON DELETE SET NULL: convertit les valeurs des clés étrangères dépendantes en valeurs NULL.

CONTRAİNTE CHECK

- ⦿ Définit une condition à laquelle chaque ligne doit satisfaire
- ⦿ Les expressions suivantes ne sont pas autorisées:
 - Appels de fonction telle que SYSDATE
 - Interrogations qui font référence à d'autres valeurs dans d'autres lignes.

```
..., salary NUMBER(2)  
    CONSTRAINT emp_salary_min  
        CHECK (salary > 0),...
```

CREATE TABLE (1)

```
CREATE TABLE employees1
( employee_id      NUMBER(6)
  CONSTRAINT emp_employee_id1 PRIMARY KEY,
  first_name       VARCHAR2(20),
  last_name        VARCHAR2(25)
  CONSTRAINT emp_last_name_nn1 NOT NULL,
  email            VARCHAR2(25)
  CONSTRAINT emp_email_nn1   NOT NULL
  CONSTRAINT emp_email_uk1   UNIQUE,
  phone_number     VARCHAR2(20),
  hire_date        DATE
  CONSTRAINT emp_hire_date_nn1 NOT NULL,
  job_id           VARCHAR2(10)
  CONSTRAINT emp_job_nn1     NOT NULL,
  salary           NUMBER(8,2)
  CONSTRAINT emp_salary_ck1  CHECK (salary>0),
  commission_pct   NUMBER(2,2),
  manager_id       NUMBER(6),
  department_id    NUMBER(4)
  CONSTRAINT emp_dept_fk1    REFERENCES
    departments (department_id));
```


CREATE TABLE (2)

```
CREATE TABLE employees2
( employee_id      NUMBER(6),
  first_name       VARCHAR2(20),
  last_name        VARCHAR2(25) NOT NULL ,
  email            VARCHAR2(25) NOT NULL ,
  phone_number     VARCHAR2(20),
  hire_date        DATE NOT NULL ,
  job_id           VARCHAR2(10) NOT NULL ,
  salary           NUMBER(8,2),
  commission_pct   NUMBER(2,2),
  manager_id       NUMBER(6),
  department_id    NUMBER(4),
CONSTRAINT emp_employee_id2 PRIMARY KEY (employee_id),
CONSTRAINT emp_email_uk2    UNIQUE (email),
CONSTRAINT emp_salary_ck2   CHECK (salary>0),
CONSTRAINT emp_dept_fk2     FOREIGN KEY (department_id)
REFERENCES departments (department_id));
```

VIOLATION DE CONTRAINTES

```
UPDATE employees  
SET    department_id = 55  
WHERE  department_id = 110;
```

```
UPDATE employees  
*  
ERROR at line 1:  
ORA-02291: integrity constraint (HR.EMP_DEPT_FK)  
violated - parent key not found
```

- Le département 55 n'existe pas

VIOLATION DE CONTRAINTES

- Vous ne pouvez pas supprimer une ligne contenant une clé primaire utilisée comme clé étrangère dans une autre table.

```
DELETE FROM departments
WHERE      department_id = 60;
```

```
DELETE FROM departments
      *
ERROR at line 1:
ORA-02292: integrity constraint (HR.EMP_DEPT_FK)
violated - child record found
```

CRÉER UNE TABLE AVEC UNE SOUS-INTERROGATION

- Créez une table et insérez des lignes en combinant l'instruction CREATE TABLE et l'option AS sous-interrogation.

```
CREATE TABLE table  
            [ (column, column...) ]  
AS subquery;
```

- Mettez en correspondance le nombre de colonnes indiqué avec le nombre de colonnes de sous-interrogation.
- Définissez les colonnes avec des noms et des valeurs par défaut.

CRÉER UNE TABLE AVEC UNE SOUS-INTERROGATION

```
CREATE TABLE dept80
```

```
AS
```

```
SELECT  employee_id, last_name,  
        salary*12 ANNSAL,  
        hire_date  
FROM    employees  
WHERE   department id = 80;
```

Table created.

```
DESCRIBE dept80
```

Name	Null?	Type
EMPLOYEE_ID		NUMBER(6)
LAST_NAME	NOT NULL	VARCHAR2(25)
ANNSAL		NUMBER
HIRE_DATE	NOT NULL	DATE

L'INSTRUCTION ALTER TABLE

- ◉ Utilisez l'instruction ALTER TABLE pour,
 - Ajouter une colonne
 - Modifier une colonne existante
 - Supprimer une colonne
 - Ajouter une contrainte
 - Supprimer une contrainte

L'INSTRUCTION ALTER TABLE

- Utilisez l'instruction ALTER TABLE pour ajouter, modifier ou supprimer des colonnes.

```
ALTER TABLE table  
ADD          (column datatype [DEFAULT expr]  
             [, column datatype]...);
```

```
ALTER TABLE table  
MODIFY       (column datatype [DEFAULT expr]  
             [, column datatype]...);
```

```
ALTER TABLE table  
DROP         (column);
```

AJOUTER UNE COLONNE

- Utilisez la clause ADD pour ajouter des colonnes.

```
ALTER TABLE dept80  
ADD      (job_id VARCHAR2(9));  
Table altered.
```

- La nouvelle colonne devient la dernière colonne.

EMPLOYEE_ID	LAST_NAME	ANNSAL	HIRE_DATE	JOB_ID
145	Russell	14000	01-OCT-96	
146	Partners	13500	05-JAN-97	
147	Errazuriz	12000	10-MAR-97	
148	Cambrault	11000	15-OCT-99	
149	Zlotkey	10500	29-JAN-00	

...

MODIFIER UNE COLONNE

- Vous pouvez changer le type de données, la taille et la valeur par défaut d'une colonne.

```
ALTER TABLE dept80  
MODIFY      (last_name VARCHAR2(30));  
Table altered.
```

- Une modification de la valeur par défaut affecte uniquement les insertions suivantes dans la table.

SUPPRIMER UNE COLONNE

- Utilisez la clause DROP COLUMN pour supprimer les colonnes de la table dont vous n'avez plus besoin.

```
ALTER TABLE dept80  
DROP COLUMN job_id;  
Table altered.
```

EMPLOYEE_ID	LAST_NAME	ANNSAL	HIRE_DATE
145	Russell	14000	01-OCT-96
146	Partners	13500	05-JAN-97
147	Errazuriz	12000	10-MAR-97
148	Cambrault	11000	15-OCT-99
149	Zlotkey	10500	29-JAN-00

AJOUTER UNE CONTRAINTE

- ◉ Utilisez l'instruction ALTER TABLE pour:
 - Ajouter ou supprimer une contrainte, sans pour autant modifier sa structure.
 - Activer ou désactiver des contraintes
 - Ajouter une contrainte NOT NULL à l'aide de la clause MODIFY.

```
ALTER TABLE <table_name>  
ADD [CONSTRAINT <constraint_name>  
type (<column_name>);
```

AJOUTER UNE CONTRAINTE

- Ajouter une contrainte FOREIGN KEY à la table EMP2 en indiquant qu'un manager doit déjà exister en tant qu'employé valide dans la table EMP2.

```
ALTER TABLE emp2  
modify employee_id Primary Key;  
Table altered.
```

```
ALTER TABLE emp2  
ADD CONSTRAINT emp_mgr_fk  
    FOREIGN KEY(manager_id)  
    REFERENCES emp2(employee_id);  
Table altered.
```

SUPPRIMER UNE CONTRAINTE

- Supprimer la contrainte manager de la table EMP2

```
ALTER TABLE emp2  
DROP CONSTRAINT emp_mgr_fk;  
Table altered.
```

- Supprimer la contrainte PRIMARY KEY sur la table DEPT2 et supprimer la contrainte FOREIGN KEY associée sur la colonne EMP2.DEPARTMENT_ID

```
ALTER TABLE dept2  
DROP PRIMARY KEY CASCADE;  
Table altered.
```

SUPPRIMER UNE TABLE

- Toutes les données de la table sont supprimées, ainsi que sa structure.
- Toutes les transactions en cours sont validées.
- Toutes les contraintes sont supprimées.
- Vous ne pouvez pas annuler l'instruction DROP TABLE.

```
DROP TABLE dept80;  
Table dropped.
```

SYNTHÈSE

- ◉ Dans ce chapitre, vous avez appris à :
 - Répertorier les principaux objets de base de données.
 - utiliser l'instruction CREATE TABLE et inclure des contraintes.
 - Modifier la structure d'une table.
 - Ajouter ou supprimer une contrainte à une table.
 - Supprimer une table.