



**Utiliser des instructions LDD pour créer et
gérer des tables**

Objectifs

A la fin de ce TP, vous pourrez :

- **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 de schéma**

Objets de base de données

Objet	Description
Table	Unité de stockage de base ; constituée de lignes
Vue	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 :

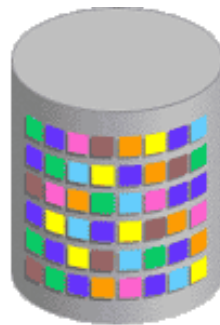
- **Doivent commencer par une lettre**
- **Doivent comporter de 1 à 30 caractères**
- **Doivent contenir uniquement les caractères A–Z, a–z, 0–9, _, \$ et #**
- **Ne doivent pas dupliquer le nom d'un autre objet appartenant au même utilisateur**
- **Ne doivent pas être un mot réservé du serveur Oracle**

Instruction CREATE TABLE

- Vous devez :
 - Posséder le privilège CREATE TABLE
 - Disposer d'une zone de stockage

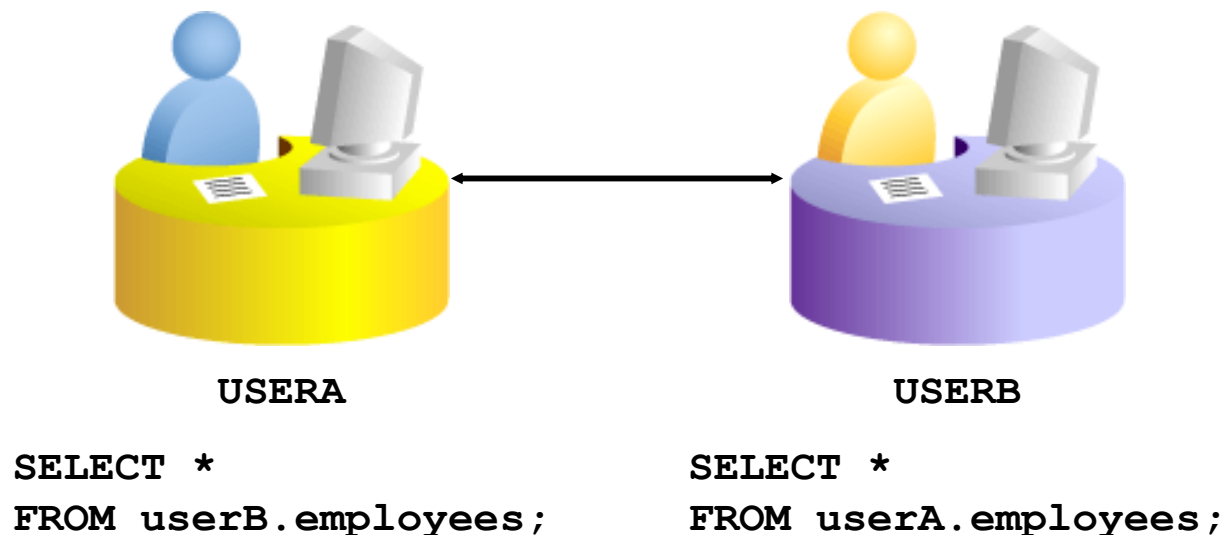
```
CREATE TABLE [schema.]table  
    (column datatype [DEFAULT expr][, ...]);
```

- 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 d'une insertion.

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

- Les valeurs littérales, les expressions et les fonctions SQL sont des valeurs légales.
- Le nom d'une autre colonne est une valeur illégale.
- Le type de données par défaut doit correspondre au type de données de la 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

Type de données	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, s</i>)	Données numériques de longueur variable
DATE	Valeurs de date et d'heure
LONG	Données de type caractère de longueur variable (jusqu'à 2 Go)
CLOB	Données de type caractère (jusqu'à 4 Go)
RAW et LONG RAW	Données binaires raw
BLOB	Données binaires (jusqu'à 4 Go)
BFILE	Données binaires stockées dans un fichier externe (jusqu'à 4 Go)
ROWID	Système de numérotation en base 64, représentant l'adresse unique d'une ligne dans sa table

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 nommer une contrainte, ou le serveur Oracle génère un nom à l'aide du format `SYS_Cn`.
- 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.
- Vous pouvez afficher une contrainte via le dictionnaire de données.

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 contrainte NOT
NULL** (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

Contrainte 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**

Contrainte 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));
```

Contrainte 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

Contrainte CHECK

- Définit une condition à laquelle chaque ligne doit satisfaire
- Les expressions suivantes ne sont pas autorisées :
 - Appels des fonctions SYSDATE et USER
 - Interrogations qui font référence à d'autres valeurs dans d'autres lignes

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

CREATE TABLE : exemple

```
CREATE TABLE employees
( employee_id      NUMBER(6)
  CONSTRAINT emp_employee_id PRIMARY KEY
, first_name       VARCHAR2(20)
, last_name        VARCHAR2(25)
  CONSTRAINT emp_last_name_nn NOT NULL
, email            VARCHAR2(25)
  CONSTRAINT emp_email_nn    NOT NULL
  CONSTRAINT emp_email_uk    UNIQUE
, phone_number     VARCHAR2(20)
, hire_date        DATE
  CONSTRAINT emp_hire_date_nn NOT NULL
, job_id           VARCHAR2(10)
  CONSTRAINT emp_job_nn      NOT NULL
, salary           NUMBER(8,2)
  CONSTRAINT emp_salary_ck   CHECK (salary>0)
, commission_pct   NUMBER(2,2)
, manager_id       NUMBER(6)
, department_id    NUMBER(4)
  CONSTRAINT emp_dept_fk     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 *subquery***.

```
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

Instruction ALTER TABLE

Utilisez l'instruction ALTER TABLE pour :

- Ajouter une colonne
- Modifier une colonne existante
- Définir une valeur par défaut pour la nouvelle colonne
- Supprimer une colonne

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 (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.
```

ON DELETE CASCADE

Supprimer les lignes enfant lorsqu'une clé parent est supprimée.

```
ALTER TABLE Emp2 ADD CONSTRAINT emp_dt_fk  
FOREIGN KEY (Department_id)  
REFERENCES departments ON DELETE CASCADE) ;  
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.
```

Désactiver des contraintes

- Exécutez la clause **DISABLE** de l'instruction **ALTER TABLE** afin de désactiver une contrainte d'intégrité.
- Appliquez l'option **CASCADE** afin de désactiver les contraintes d'intégrité dépendantes.

```
ALTER TABLE emp2  
DISABLE CONSTRAINT emp_dt_fk;  
Table altered.
```

Activer des contraintes

- Utilisez la clause **ENABLE** pour activer une contrainte d'intégrité actuellement désactivée dans la définition de la table.

```
ALTER TABLE      emp2
ENABLE CONSTRAINT emp_dt_fk;
Table altered.
```

- Un index **UNIQUE** est automatiquement créé si vous activez une clé **UNIQUE** ou une contrainte **PRIMARY KEY**.

Supprimer une table

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

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

Synthèse

Dans ce TP, vous avez appris à utiliser l'instruction `CREATE TABLE` pour créer une table et inclure des contraintes :

- **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 de schéma**