

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

# Objets de base de données

Objet	Description
Table	Unité élémentaire de stockage, constituée de lignes
Vue	Représente de façon logique des sous-ensembles de données issus d'une ou de plusieurs tables
Séquence	Génère des valeurs numériques
Index	Améliore les performances de certaines interrogations
Synonyme	Attribue un autre nom à un objet

# Règles d'appellation

Les noms de table et de colonne :

- doivent commencer par une lettre
- doivent comporter entre 1 et 30 caractères
- admettent uniquement les caractères A–Z, a–z, 0–9, \_, \$ et #
- ne peuvent pas être identiques au nom d'un autre objet appartenant au même utilisateur
- ne doivent pas être des mots réservés au serveur Oracle

# Instruction CREATE TABLE

- Vous devez disposer des éléments suivants :
  - le privilège `CREATE TABLE`
  - une zone de stockage

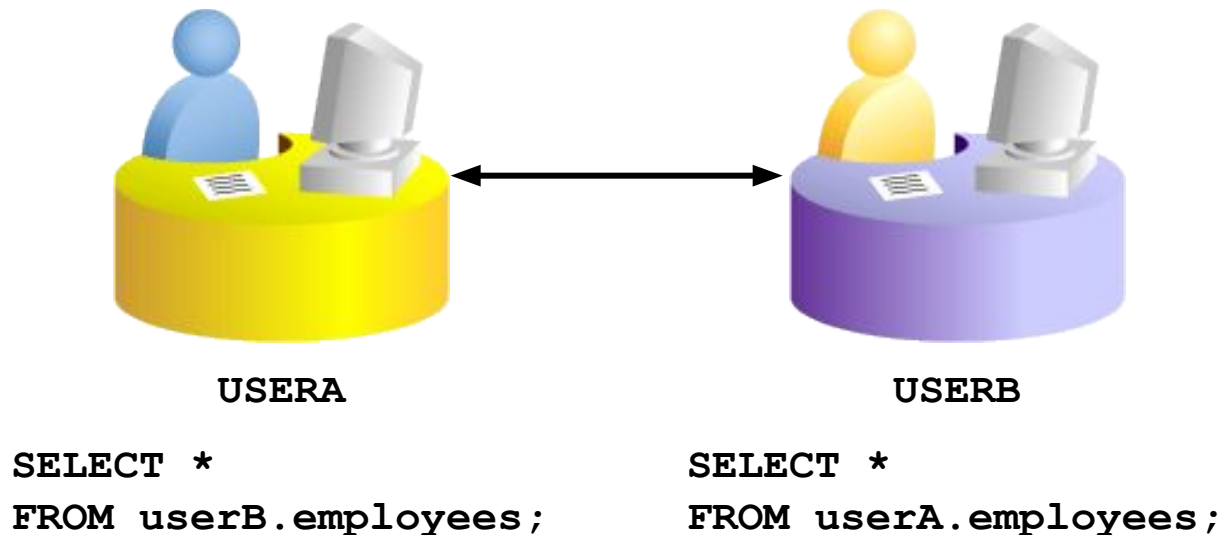
```
CREATE TABLE [schema.]table  
              (column datatype [DEFAULT expr][, ...]);
```

- Vous indiquez :
  - le nom de la table
  - le nom de colonne, le type de données des colonnes et la taille de colonne



# Référencer les tables d'un autre utilisateur

- Les tables appartenant à d'autres utilisateurs ne figurent pas dans le schéma de l'utilisateur en cours.
- Vous devez ajouter le nom du propriétaire comme préfixe à ces tables.



## Option DEFAULT

- Lors d'une insertion, indiquez une valeur de colonne par défaut.

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

- Valeurs autorisées : Valeurs littérales, expressions ou fonctions SQL.
- Valeurs non autorisées : Nom d'une autre colonne ou pseudo-colonne.
- Le type de données par défaut doit correspondre à celui de la colonne.

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

```
CREATE TABLE succeeded.
```

# Créer des tables

- Créez la table :

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

```
CREATE TABLE succeeded.
```

- Vérifiez la création de la table :

```
DESCRIBE dept
```

NAME	NULL	TYPE
DEPTNO		NUMBER(2)
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)
CREATE_DATE		DATE

4 rows selected

# Types de données

Type de données	Description
VARCHAR2 ( <i>size</i> )	Données alphanumériques de longueur variable
CHAR ( <i>size</i> )	Données alphanumériques de longueur fixe
NUMBER ( <i>p</i> , <i>s</i> )	Données numériques de longueur variable
DATE	Valeur date-heure
LONG	Données alphanumériques de longueur variable (jusqu'à 2 Go)
CLOB	Données alphanumériques (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	Nombre en base 64 représentant l'adresse unique d'une ligne dans sa table



# Types de données date-heure

Vous pouvez utiliser plusieurs types de données date-heure :

Type de données	Description
TIMESTAMP	Date avec partie décimale des secondes
INTERVAL YEAR TO MONTH	Intervalle exprimé en années et en mois
INTERVAL DAY TO SECOND	Intervalle exprimé en jours, heures, minutes et secondes



# Inclure des contraintes

- Les contraintes appliquent des règles de niveau table.
- Elles empêchent la suppression d'une table dotée de dépendances.
- Les types de contrainte valides sont les suivants :
  - NOT NULL
  - UNIQUE
  - PRIMARY KEY
  - FOREIGN KEY
  - CHECK



# Règles relatives aux contraintes

- Vous pouvez nommer une contrainte ou laisser le serveur Oracle générer un nom au format `SYS_Cn`.
- Vous pouvez créer une contrainte à l'un des stades suivants :
  - lors de la création de la table
  - après la création de la table
- Vous pouvez définir une contrainte de niveau colonne ou table.
- Vous pouvez visualiser une contrainte dans le dictionnaire de données.

# Définir des contraintes

- Syntaxe :

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

- Syntaxe d'une contrainte de niveau colonne :

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

- Syntaxe d'une contrainte de niveau table :

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

# Définir des contraintes

- Exemple de contrainte de niveau table :

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

1

- Exemple de contrainte de 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

Elle garantit que les valeurs NULL ne sont pas autorisées pour la colonne :

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	COMMISSION_PCT	DEPARTMENT_ID	EMAIL	PHONE_NUMBER	HIRE_DATE
100	Steven	King	24000	(null)	90	SKING	515.123.4567	17-JUN-87
101	Neena	Kochhar	17000	(null)	90	NKOCHHAR	515.123.4568	21-SEP-89
102	Lex	De Haan	17000	(null)	90	LDEHAAN	515.123.4569	13-JAN-93
103	Alexander	Hunold	9000	(null)	60	AHUNOLD	590.423.4567	03-JAN-90
104	Bruce	Ernst	6000	(null)	60	BERNST	590.423.4568	21-MAY-91
107	Diana	Lorentz	4200	(null)	60	DLORENTZ	590.423.5567	07-FEB-99
124	Kevin	Mourgos	5800	(null)	50	KMOURGOS	650.123.5234	16-NOV-99
141	Trenna	Rajs	3500	(null)	50	TRAJS	650.121.8009	17-OCT-95
142	Curtis	Davies	3100	(null)	50	CDAVIES	650.121.2994	29-JAN-97
143	Randall	Matos	2600	(null)	50	RMATOS	650.121.2874	15-MAR-98
144	Peter	Vargas	2500	(null)	50	PVARGAS	650.121.2004	09-JUL-98
149	Eleni	Zlotkey	10500	0.2	80	EZLOTKEY	011.44.1344.429018	29-JAN-00
174	Ellen	Abel	11000	0.3	80	EABEL	011.44.1644.429267	11-MAY-96
176	Jonathon	Taylor	8600	0.2	80	JTAYLOR	011.44.1644.429265	24-MAR-98
178	Kimberely	Grant	7000	0.15	(null)	KGRANT	011.44.1644.429263	24-MAY-99
200	Jennifer	Whalen	4400	(null)	10	JWHALEN	515.123.4444	17-SEP-87
201	Michael	Hartstein	13000	(null)	20	MHARTSTE	515.123.5555	17-FEB-96
202	Pat	Fay	6000	(null)	20	PFAY	603.123.6666	17-AUG-97
205	Shelley	Higgins	12000	(null)	110	SHIGGINS	515.123.8080	07-JUN-94
206	William	Gietz	8300	(null)	110	WGIEZT	515.123.8181	07-JUN-94

↑  
**Contrainte NOT NULL**  
(La clé primaire applique une contrainte NOT NULL.)

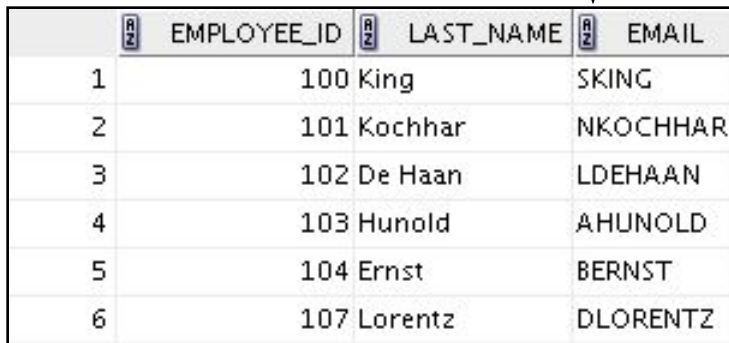
↑  
**Contrainte NOT NULL**




↑  
**Absence de contrainte NOT NULL**  
(N'importe quelle ligne peut contenir une valeur NULL pour cette colonne.)

# Contrainte UNIQUE

## EMPLOYEES

Contrainte UNIQUE



		EMPLOYEE_ID		LAST_NAME		EMAIL
1		100		King		SKING
2		101		Kochhar		NKOCHHAR
3		102		De Haan		LDEHAAN
4		103		Hunold		AHUNOLD
5		104		Ernst		BERNST
6		107		Lorentz		DLORENTZ

...



INSERT INTO

208	SMITH	JSMITH
-----	-------	--------

← Autorisé

209	SMITH	JSMITH
-----	-------	--------

← Non autorisé : existe déjà

# Contrainte UNIQUE

Elle est 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
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700

**Non autorisé  
(valeur NULL)** 

 **INSERT INTO**


(null)	Public Accounting	124	2500
50	Finance	124	1500




**Non autorisé  
(50 existe déjà)** 

# Contrainte FOREIGN KEY

## DEPARTMENTS

**PRIMARY  
KEY**





	 DEPARTMENT_ID	 DEPARTMENT_NAME	 MANAGER_ID	 LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500

...

## EMPLOYEES



	 EMPLOYEE_ID	 LAST_NAME	 DEPARTMENT_ID
1	200	Whalen	10
2	201	Hartstein	20
3	202	Fay	20
4	205	Higgins	110
5	206	Gietz	110

**FOREIGN KEY**

...



**INSERT INTO**

200 Ford	9
200 Ford	60

**Non autorisé  
(9 n'existe pas)**

**Autorisé**

# Contrainte FOREIGN KEY

Elle est 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 table.
- REFERENCES : Identifie la table et la colonne dans la table parent.
- ON DELETE CASCADE : Supprime les lignes dépendantes dans la table enfant en cas de suppression d'une ligne dans la table parent.
- ON DELETE SET NULL : Convertit les valeurs de clé étrangère dépendantes en valeurs NULL.

# Contrainte CHECK

- Elle définit une condition à laquelle chaque ligne doit satisfaire.
- Les expressions suivantes ne sont pas autorisées :
  - Références aux pseudo-colonnes CURRVAL, NEXTVAL, LEVEL et ROWNUM
  - Appels de fonctions SYSDATE, UID, USER et USERENV
  - Interrogations faisant 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)
CONSTRAINT emp_manager_fk REFERENCES
employees (employee_id)
, 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;
```

Error starting at line 1 in command:

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

Error report:

```
SQL Error: ORA-02291: integrity constraint (ORA1.EMP_DEPT_FK) violated - parent key not found
02291. 00000 - "integrity constraint (%s.%s) violated - parent key not found"
*Cause:      A foreign key value has no matching primary key value.
```

Le département 55 n'existe pas.

# Violation de contraintes

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

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

Error starting at line 1 in command:

DELETE FROM departments

WHERE department\_id = 60

Error report:

SQL Error: ORA-02292: integrity constraint (ORA1.JHIST\_DEPT\_FK) violated - child record found  
02292. 00000 - "integrity constraint (%s.%s) violated - child record found"

\*Cause: attempted to delete a parent key value that had a foreign  
dependency.

\*Action: delete dependencies first then parent or disable constraint.



# Créer une table à l'aide d'une sous-interrogation

- Créez une table et insérez-y des lignes en associant l'instruction `CREATE TABLE` et l'option `AS subquery`.

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

- Faites correspondre le nombre de colonnes indiquées et le nombre de colonnes de la sous-interrogation.
- Définissez les colonnes avec des noms et des valeurs par défaut.

# Créer une table à l'aide d'une sous-interrogation

```
CREATE TABLE dept80
AS
SELECT  employee_id, last_name,
        salary*12 ANNSAL,
        hire_date
FROM    employees
WHERE   department_id = 80;
```

```
CREATE TABLE succeeded.
```

```
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 effectuer les opérations suivantes :

- Ajouter une nouvelle colonne
- Modifier la définition d'une colonne existante
- Définir une valeur par défaut pour la nouvelle colonne
- Supprimer une colonne
- Renommer une colonne
- Mettre une table en lecture seule

# Tables en lecture seule

Utilisez l'instruction `ALTER TABLE` pour effectuer les opérations suivantes :

- Placer une table en mode lecture seule, afin d'empêcher les opérations LDD ou LMD pendant la phase de maintenance.
- Remettre la table en mode lecture/écriture.

```
ALTER TABLE employees READ ONLY;
```

```
-- perform table maintenance and then
```

```
-- return table back to read/write mode
```

```
ALTER TABLE employees READ WRITE;
```

# Instruction de suppression de table

- Place une table dans la corbeille.
- Supprime entièrement la table et l'ensemble de ses données si la clause `PURGE` est indiquée.
- Invalide les objets dépendants et supprime les privilèges objet associés à la table.

```
DROP TABLE dept80;
```

```
DROP TABLE dept80 succeeded.
```