

# **REQUÊTES AVEC JOINTURES**

**+**

# **SOUS REQUÊTE**

BTS DAKHLA

DAHAR RACHID

# RÉFÉRENCE BIBLIOGRAPHIQUE

- Prof. Saliha yaakoub

Nous utilisons les tables ci-dessous durant l'explication :

**Table emp**

	empno	ename	job	mgr	hiredate	sal	comm	dept
1	7369	SMITH	CLERK	7902	1980-12-17 00:00:00.000	800.00	NULL	20
2	7499	ALLEN	SALESMAN	7698	1981-02-20 00:00:00.000	1600.00	300.00	30
3	7521	WARD	SALESMAN	7698	1981-02-22 00:00:00.000	1250.00	500.00	30
4	7566	JONES	MANAGER	7839	1981-04-02 00:00:00.000	2975.00	NULL	20
5	7654	MARTIN	SALESMAN	7698	1981-09-28 00:00:00.000	1250.00	1400.00	30
6	7698	BLAKE	MANAGER	7839	1981-05-01 00:00:00.000	2850.00	NULL	30
7	7782	CLARK	MANAGER	7839	1981-06-09 00:00:00.000	2450.00	NULL	10
8	7788	SCOTT	ANALYST	7566	1982-12-09 00:00:00.000	3000.00	NULL	20
9	7839	KING	PRESIDENT	NULL	1981-11-17 00:00:00.000	5000.00	NULL	10
10	7844	TURNER	SALESMAN	7698	1981-09-08 00:00:00.000	1500.00	0.00	30
11	7876	ADAMS	CLERK	7788	1983-01-12 00:00:00.000	1100.00	NULL	20
12	7900	JAMES	CLERK	7698	1981-12-03 00:00:00.000	950.00	NULL	30
13	7902	FORD	ANALYST	7566	1981-12-03 00:00:00.000	3000.00	NULL	20
14	7934	MILLER	CLERK	7782	1982-01-23 00:00:00.000	1300.00	NULL	10
15	7935	MOHAMED	PRESIDENT	NULL	1981-06-15 00:00:00.000	5000.00	NULL	NULL
16	7940	DRISS	MANAGER	7935	1982-07-20 00:00:00.000	3500.00	NULL	NULL
17	7950	HICHAM	ANALYST	7940	1981-09-26 00:00:00.000	5000.00	NULL	NULL
18	7951	IMAD	SALESMAN	7940	1982-10-27 00:00:00.000	5000.00	200.00	NULL
19	7952	AHMED	CLERK	7950	1983-11-29 00:00:00.000	5000.00	NULL	NULL

**Table dept**

	deptno	dname	loc
1	10	ACCOUNTING	NEW YORK
2	20	RESEARCH	DALLAS
3	30	SALES	CHICAGO
4	40	OPERATIONS	BOSTON

# PRODUIT CARTÉSIEN

Le produit cartésien est une requête de sélection qui met en jeu plusieurs tables.

Pour deux tables, la sélection consiste à afficher:

- la première ligne de la première table avec toutes les lignes de la deuxième table,
- la deuxième ligne de la première table avec toutes les lignes de la deuxième table et ainsi de suite.

Ce type de sélection implique beaucoup de redondances.

**SELECT \* FROM dept,emp;**

Va nous donner comme résultat 76 lignes = (4\*19)

Il y a beaucoup de redondance.

deptno	dname	loc	emppno	ename	job	mgr	hiredate	sal	comm	dept
1 10	ACCOUNTING	NEW YORK	7369	SMITH	CLERK	7902	1980-12-17 00:00:00.000	800.00	NULL	20
2 10	ACCOUNTING	NEW YORK	7499	ALLEN	SALESMAN	7898	1981-01-20 00:00:00.000	1600.00	300.00	20
3 10	ACCOUNTING	NEW YORK	7521	WARD	SALESMAN	7898	1981-02-01 00:00:00.000	1250.00	500.00	30
4 10	ACCOUNTING	NEW YORK	7566	JONES	MANAGER	7839	1981-04-02 00:00:00.000	2975.00	NULL	20
5 10	ACCOUNTING	NEW YORK	7654	MARTIN	SALESMAN	7898	1981-09-28 00:00:00.000	1250.00	1400.00	30
6 10	ACCOUNTING	NEW YORK	7698	BLAKE	MANAGER	7839	1981-05-01 00:00:00.000	2850.00	NULL	30
7 10	ACCOUNTING	NEW YORK	7782	CLARK	MANAGER	7839	1981-06-09 00:00:00.000	2450.00	NULL	10
8 10	ACCOUNTING	NEW YORK	7888	SCOTT	ANALYST	7566	1982-12-09 00:00:00.000	3000.00	NULL	20
9 10	ACCOUNTING	NEW YORK	7839	KING	PRESIDENT	NULL	1981-11-17 00:00:00.000	5000.00	NULL	10
10 10	ACCOUNTING	NEW YORK	7844	TURNER	SALESMAN	7698	1981-09-08 00:00:00.000	1500.00	0.00	30
11 10	ACCOUNTING	NEW YORK	7876	ADAMS	CLERK	7788	1983-01-12 00:00:00.000	1100.00	NULL	20
12 10	ACCOUNTING	NEW YORK	7900	JAMES	CLERK	7698	1981-12-03 00:00:00.000	950.00	NULL	30
13 10	ACCOUNTING	NEW YORK	7902	FORD	ANALYST	7566	1981-12-03 00:00:00.000	3000.00	NULL	20
14 10	ACCOUNTING	NEW YORK	7934	MILLER	CLERK	7788	1982-01-23 00:00:00.000	1300.00	NULL	10
15 10	ACCOUNTING	NEW YORK	7940	OHMAD	PRESIDENT	NULL	1982-01-15 00:00:00.000	5000.00	NULL	NULL
16 10	ACCOUNTING	NEW YORK	7940	DRISS	MANAGER	7935	1982-07-06 00:00:00.000	3000.00	NULL	NULL
17 10	ACCOUNTING	NEW YORK	7950	HICHAM	ANALYST	7940	1981-09-26 00:00:00.000	5000.00	NULL	NULL
18 10	ACCOUNTING	NEW YORK	7951	IMAD	SALESMAN	7940	1982-10-27 00:00:00.000	5000.00	200.00	NULL
19 10	ACCOUNTING	NEW YORK	7952	AHMED	CLERK	7950	1983-11-29 00:00:00.000	5000.00	NULL	NULL
20 20	RESEARCH	DALLAS	7369	SMITH	CLERK	7902	1980-12-17 00:00:00.000	800.00	NULL	20
21 20	RESEARCH	DALLAS	7499	ALLEN	SALESMAN	7698	1981-02-20 00:00:00.000	1600.00	300.00	30
22 20	RESEARCH	DALLAS	7521	WARD	SALESMAN	7698	1981-02-22 00:00:00.000	1250.00	500.00	30
23 20	RESEARCH	DALLAS	7566	JONES	MANAGER	7839	1981-04-02 00:00:00.000	2975.00	NULL	20
24 20	RESEARCH	DALLAS	7654	MARTIN	SALESMAN	7698	1981-09-28 00:00:00.000	1250.00	1400.00	30
25 20	RESEARCH	DALLAS	7698	BLAKE	MANAGER	7839	1981-05-01 00:00:00.000	2850.00	NULL	30
26 20	RESEARCH	DALLAS	7782	CLARK	MANAGER	7839	1981-06-09 00:00:00.000	2450.00	NULL	10
27 20	RESEARCH	DALLAS	7788	SCOTT	ANALYST	7566	1982-12-09 00:00:00.000	3000.00	NULL	20
28 20	RESEARCH	DALLAS	7836	KING	PRESIDENT	NULL	1982-01-15 00:00:00.000	5000.00	NULL	10
29 20	RESEARCH	DALLAS	7844	TURNER	SALESMAN	7698	1981-09-08 00:00:00.000	1500.00	0.00	30
30 20	RESEARCH	DALLAS	7876	ADAMS	CLERK	7788	1983-01-12 00:00:00.000	1100.00	NULL	20
31 20	RESEARCH	DALLAS	7900	JAMES	CLERK	7698	1981-12-03 00:00:00.000	950.00	NULL	30
32 20	RESEARCH	DALLAS	7902	FORD	ANALYST	7566	1981-12-03 00:00:00.000	3000.00	NULL	20
33 20	RESEARCH	DALLAS	7934	MILLER	CLERK	7788	1982-01-23 00:00:00.000	1300.00	NULL	10
34 20	RESEARCH	DALLAS	7935	MOHAMED	PRESIDENT	NULL	1981-06-15 00:00:00.000	5000.00	NULL	NULL
35 20	RESEARCH	DALLAS	7940	DRISS	MANAGER	7935	1982-07-20 00:00:00.000	3500.00	NULL	NULL
36 20	RESEARCH	DALLAS	7950	HICHAM	ANALYST	7940	1981-09-26 00:00:00.000	5000.00	NULL	NULL
37 20	RESEARCH	DALLAS	7951	IMAD	SALESMAN	7940	1982-10-27 00:00:00.000	5000.00	200.00	NULL
38 20	RESEARCH	DALLAS	7952	AHMED	CLERK	7950	1983-11-29 00:00:00.000	5000.00	NULL	NULL
39 30	SALES	CHICAGO	7369	SMITH	CLERK	7902	1980-12-17 00:00:00.000	800.00	NULL	20
40 30	SALES	CHICAGO	7499	ALLEN	SALESMAN	7698	1981-02-20 00:00:00.000	1600.00	300.00	30
41 30	SALES	CHICAGO	7521	WARD	SALESMAN	7698	1981-02-22 00:00:00.000	1250.00	500.00	30
42 30	SALES	CHICAGO	7566	JONES	MANAGER	7839	1981-04-02 00:00:00.000	2975.00	NULL	20
43 30	SALES	CHICAGO	7654	MARTIN	SALESMAN	7698	1981-09-28 00:00:00.000	1250.00	1400.00	30
44 30	SALES	CHICAGO	7698	BLAKE	MANAGER	7839	1981-05-01 00:00:00.000	2850.00	NULL	30
45 30	SALES	CHICAGO	7782	CLARK	MANAGER	7839	1981-06-09 00:00:00.000	2450.00	NULL	10
46 30	SALES	CHICAGO	7788	SCOTT	ANALYST	7566	1982-12-09 00:00:00.000	3000.00	NULL	20
47 30	SALES	CHICAGO	7839	KING	PRESIDENT	NULL	1981-11-17 00:00:00.000	5000.00	NULL	10
48 30	SALES	CHICAGO	7844	TURNER	SALESMAN	7698	1981-09-08 00:00:00.000	1500.00	0.00	30
49 30	SALES	CHICAGO	7876	ADAMS	CLERK	7788	1983-01-12 00:00:00.000	1100.00	NULL	20
50 30	SALES	CHICAGO	7900	JAMES	CLERK	7698	1981-12-03 00:00:00.000	950.00	NULL	30
51 30	SALES	CHICAGO	7902	FORD	ANALYST	7566	1982-12-03 00:00:00.000	3000.00	NULL	20
52 30	SALES	CHICAGO	7934	MILLER	CLERK	7788	1982-01-23 00:00:00.000	1300.00	NULL	10
53 30	SALES	CHICAGO	7935	MOHAMED	PRESIDENT	NULL	1981-06-15 00:00:00.000	5000.00	NULL	NULL
54 30	SALES	CHICAGO	7940	DRISS	MANAGER	7935	1982-10-20 00:00:00.000	3500.00	NULL	NULL
55 30	SALES	CHICAGO	7950	HICHAM	ANALYST	7940	1981-09-26 00:00:00.000	5000.00	NULL	NULL
56 30	SALES	CHICAGO	7951	IMAD	SALESMAN	7940	1982-10-27 00:00:00.000	5000.00	200.00	NULL
57 30	SALES	CHICAGO	7952	AHMED	CLERK	7950	1983-11-29 00:00:00.000	5000.00	NULL	NULL
58 40	OPERATIONS	BOSTON	7369	SMITH	CLERK	7902	1980-12-17 00:00:00.000	800.00	NULL	20
59 40	OPERATIONS	BOSTON	7499	ALLEN	SALESMAN	7698	1981-02-20 00:00:00.000	1600.00	300.00	30
60 40	OPERATIONS	BOSTON	7521	WARD	SALESMAN	7698	1981-02-22 00:00:00.000	1250.00	500.00	30
61 40	OPERATIONS	BOSTON	7566	JONES	MANAGER	7839	1981-04-02 00:00:00.000	2975.00	NULL	20
62 40	OPERATIONS	BOSTON	7654	MARTIN	SALESMAN	7698	1981-09-28 00:00:00.000	1250.00	1400.00	30
63 40	OPERATIONS	BOSTON	7698	BLAKE	MANAGER	7839	1981-05-01 00:00:00.000	2850.00	NULL	30
64 40	OPERATIONS	BOSTON	7782	CLARK	MANAGER	7839	1981-06-09 00:00:00.000	2450.00	NULL	10
65 40	OPERATIONS	BOSTON	7788	SCOTT	ANALYST	7566	1982-12-09 00:00:00.000	3000.00	NULL	20
66 40	OPERATIONS	BOSTON	7839	KING	PRESIDENT	NULL	1981-11-17 00:00:00.000	5000.00	NULL	10
67 40	OPERATIONS	BOSTON	7844	TURNER	SALESMAN	7698	1981-09-08 00:00:00.000	1500.00	0.00	30
68 40	OPERATIONS	BOSTON	7876	ADAMS	CLERK	7788	1983-01-12 00:00:00.000	1100.00	NULL	20
69 40	OPERATIONS	BOSTON	7900	JAMES	CLERK	7698	1981-12-03 00:00:00.000	950.00	NULL	30
70 40	OPERATIONS	BOSTON	7902	FORD	ANALYST	7566	1981-12-03 00:00:00.000	3000.00	NULL	20
71 40	OPERATIONS	BOSTON	7934	MILLER	CLERK	7788	1982-01-23 00:00:00.000	1300.00	NULL	10
72 40	OPERATIONS	BOSTON	7935	MOHAMED	PRESIDENT	NULL	1981-06-15 00:00:00.000	5000.00	NULL	NULL
73 40	OPERATIONS	BOSTON	7940	DRISS	MANAGER	7935	1982-07-20 00:00:00.000	3500.00	NULL	NULL
74 40	OPERATIONS	BOSTON	7950	HICHAM	ANALYST	7940	1981-09-26 00:00:00.000	5000.00	NULL	NULL
75 40	OPERATIONS	BOSTON	7951	IMAD	SALESMAN	7940	1982-10-27 00:00:00.000	5000.00	200.00	NULL
76 40	OPERATIONS	BOSTON	7952	AHMED	CLERK	7950	1983-11-29 00:00:00.000	5000.00	NULL	NULL

# **JOINTURE**

**Une jointure** est une opération relationnelle qui sert à chercher des lignes (ou des enregistrements) à partir de deux ou plusieurs tables disposant d'un ensemble de valeurs communes, en général les clés primaires.

Il existe plusieurs types de jointure que nous allons aborder au fur et à mesure. Parmi ces jointures, nous avons la jointure interne appelée aussi jointure simple

**Une jointure simple** consiste en un produit cartésien avec un **INNER JOIN** faisant ainsi une restriction sur les lignes. La restriction est faîte sur l'égalité de la valeur de deux attributs (cas de deux tables) qui sont **la valeur d'une clé primaire est égale à la valeur d'une clé étrangère.**

Les jointures se font au niveau de la clause **FROM**

```
SELECT ename , job , dname FROM emp e
INNER JOIN dept d ON e.dept = d.deptno
```

```
SELECT ename , job , dname FROM emp e
JOIN dept d ON e.dept = d.deptno
WHERE comm is NULL
```

Lorsqu'un **attribut sélectionné** est présent dans plus d'une table alors il faut le précéder du nom de la table à partir de laquelle on désire l'extraire. Si le nom de la table n'est pas précisé cela va renvoyer une erreur

	ename	job	dname
1	SMITH	CLERK	RESEARCH
2	ALLEN	SALESMAN	SALES
3	WARD	SALESMAN	SALES
4	JONES	MANAGER	RESEARCH
5	MARTIN	SALESMAN	SALES
6	BLAKE	MANAGER	SALES
7	CLARK	MANAGER	ACCOUNTING
8	SCOTT	ANALYST	RESEARCH
9	KING	PRESIDENT	ACCOUNTING
10	TURNER	SALESMAN	SALES
11	ADAMS	CLERK	RESEARCH
12	JAMES	CLERK	SALES
13	FORD	ANALYST	RESEARCH
14	MILLER	CLERK	ACCOUNTING

	ename	job	dname
1	SMITH	CLERK	RESEARCH
2	JONES	MANAGER	RESEARCH
3	BLAKE	MANAGER	SALES
4	CLARK	MANAGER	ACCOUNTING
5	SCOTT	ANALYST	RESEARCH
6	KING	PRESIDENT	ACCOUNTING
7	ADAMS	CLERK	RESEARCH
8	JAMES	CLERK	SALES
9	FORD	ANALYST	RESEARCH
10	MILLER	CLERK	ACCOUNTING

Une requête avec jointure interne INNER JOIN, va ramener des résultats des **deux tables emp et dept** uniquement s'il y égalité entre la **clé étrangère dept de la table emp** et la **clé primaire deptno de la table dept**.

Selon le contenu initial des tables, la requête va ramener exactement 14 enregistrements (lignes)

Les employés qui n'ont pas de département ne seront pas ramenés. Les départements qui n'ont pas d'employés ne seront pas ramenées

# UTILISATION DES ALIAS

Vous pouvez donner un alias aux noms de tables afin de **faciliter la référence aux tables**. Cependant si un alias est donné, alors il faudra utiliser l'alias à la place du nom de la table

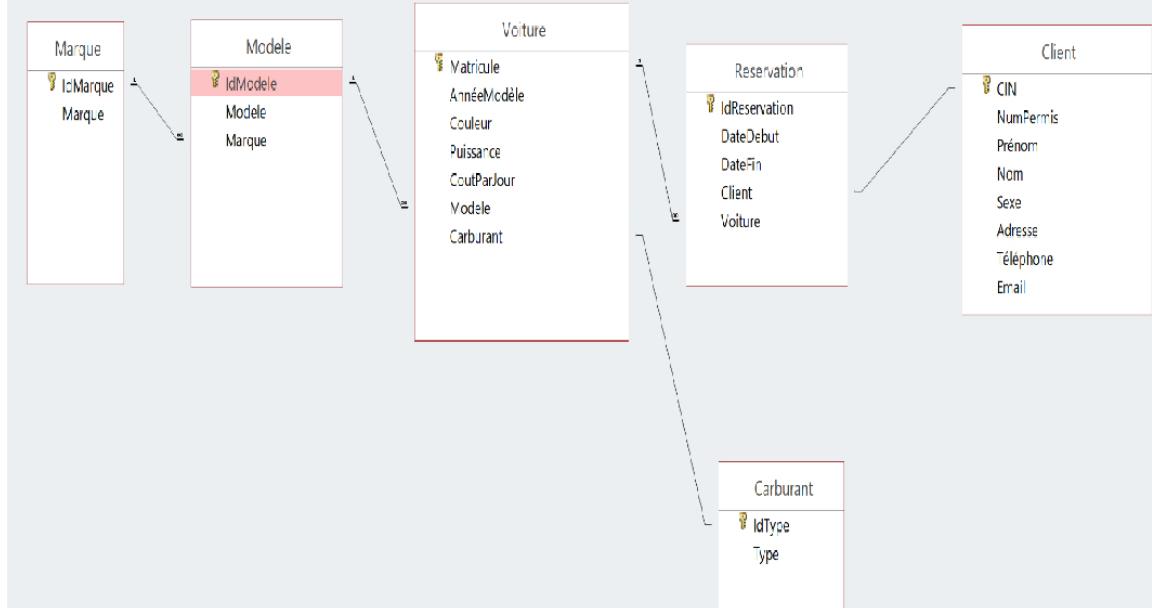
L'exemple suivant va renvoyer une erreur

```
SELECT ename , job , dname FROM emp e JOIN dept d  
ON e.dept = dept.deptno  
WHERE comm is NULL
```

# JOINTURES ET TABLE DE LIEN

Soit le modèle relationnel suivant

**Requête :** Afficher la liste des voitures diesel produites entre 2016 et 2019 de la marque Seat , triées par ordre croissant de puissance



```
SELECT matricule , AnneeModele , puissance , c.Type , m.Marque FROM Voiture v
JOIN Carburant c ON v.Carburant = c.idType
JOIN Modele mo ON v.Modele = mo.idModele
JOIN Marque m ON mo.Marque = m.idMarque
WHERE m.Marque = 'SEAT' AND c.Type = 'Diesel'
AND AnneeModele BETWEEN 2016 AND 2019
```

# AUTOJOINTURE

Une auto-jointure est une jointure d'une table avec elle-même

Requête : l'identité des employés placés sous la responsabilité du manager BLAKE

```
SELECT e1.empno , e1.ename FROM  
emp e1  
JOIN emp e2 ON e1.mgr = e2.empno  
WHERE e2.ename = 'BLAKE'
```

	empno	ename	job	mgr	hiredate	sal	comm	dept
1	7369	SMITH	CLERK	7902	1980-12-17 00:00:00.000	800.00	NULL	20
2	7499	ALLEN	SALESMAN	7698	1981-02-20 00:00:00.000	1600.00	300.00	30
3	7521	WARD	SALESMAN	7698	1981-02-22 00:00:00.000	1250.00	500.00	30
4	7566	JONES	MANAGER	7839	1981-04-02 00:00:00.000	2975.00	NULL	20
5	7654	MARTIN	SALESMAN	7698	1981-09-28 00:00:00.000	1250.00	1400.00	30
6	7698	BLAKE	MANAGER	7839	1981-05-01 00:00:00.000	2850.00	NULL	30
7	7782	CLARK	MANAGER	7839	1981-06-09 00:00:00.000	2450.00	NULL	10
8	7788	SCOTT	ANALYST	7566	1982-12-09 00:00:00.000	3000.00	NULL	20
9	7839	KING	PRESIDENT	NULL	1981-11-17 00:00:00.000	5000.00	NULL	10
10	7844	TURNER	SALESMAN	7698	1981-09-08 00:00:00.000	1500.00	0.00	30
11	7876	ADAMS	CLERK	7788	1983-01-12 00:00:00.000	1100.00	NULL	20
12	7900	JAMES	CLERK	7698	1981-12-03 00:00:00.000	950.00	NULL	30
13	7902	FORD	ANALYST	7566	1981-12-03 00:00:00.000	3000.00	NULL	20
14	7934	MILLER	CLERK	7782	1982-01-23 00:00:00.000	1300.00	NULL	10
15	7935	MOHAMED	PRESIDENT	NULL	1981-06-15 00:00:00.000	5000.00	NULL	NULL
16	7940	DRISS	MANAGER	7935	1982-07-20 00:00:00.000	3500.00	NULL	NULL
17	7950	HICHAM	ANALYST	7940	1981-09-26 00:00:00.000	5000.00	NULL	NULL
18	7951	IMAD	SALESMAN	7940	1982-10-27 00:00:00.000	5000.00	200.00	NULL
19	7952	AHMED	CLERK	7950	1983-11-29 00:00:00.000	5000.00	NULL	NULL

	empno	ename
1	7499	ALLEN
2	7521	WARD
3	7654	MARTIN
4	7844	TURNER
5	7900	JAMES

# JOINTURE EXTERNE LEFT OUTER JOIN

Dans la jointure externe gauche, des enregistrements de table à gauche de la jointure seront ramenés même si ceux-ci n'ont pas d'occurrences dans l'autre table.

**LEFT JOIN** = INNER JOIN + tout enregistrement supplémentaire dans la table de gauche

```
SELECT ename , job , dname FROM emp e  
LEFT JOIN dept d ON e.dept = d.deptno
```

	ename	job	dname
1	SMITH	CLERK	RESEARCH
2	ALLEN	SALESMAN	SALES
3	WARD	SALESMAN	SALES
4	JONES	MANAGER	RESEARCH
5	MARTIN	SALESMAN	SALES
6	BLAKE	MANAGER	SALES
7	CLARK	MANAGER	ACCOUNTING
8	SCOTT	ANALYST	RESEARCH
9	KING	PRESIDENT	ACCOUNTING
10	TURNER	SALESMAN	SALES
11	ADAMS	CLERK	RESEARCH
12	JAMES	CLERK	SALES
13	FORD	ANALYST	RESEARCH
14	MILLER	CLERK	ACCOUNTING
15	MOHAMED	PRESIDENT	NULL
16	DRISS	MANAGER	NULL
17	HICHAM	ANALYST	NULL
18	IMAD	SALESMAN	NULL
19	AHMED	CLERK	NULL

# JOINTURE EXTERNE **RIGHT OUTER JOIN**

Dans la jointure externe droite, des enregistrements de table à droite de la jointure seront ramenés même si ceux-ci n'ont pas d'occurrences dans l'autre table.

**RIGHT JOIN** = INNER JOIN + tout enregistrement supplémentaire dans **la table de droite**

```
SELECT ename , job , dname FROM emp e  
RIGHT JOIN dept d ON e.deptno = d.deptno
```

	ename	job	dname
1	CLARK	MANAGER	ACCOUNTING
2	KING	PRESIDENT	ACCOUNTING
3	MILLER	CLERK	ACCOUNTING
4	SMITH	CLERK	RESEARCH
5	JONES	MANAGER	RESEARCH
6	SCOTT	ANALYST	RESEARCH
7	ADAMS	CLERK	RESEARCH
8	FORD	ANALYST	RESEARCH
9	ALLEN	SALESMAN	SALES
10	WARD	SALESMAN	SALES
11	MARTIN	SALESMAN	SALES
12	BLAKE	MANAGER	SALES
13	TURNER	SALESMAN	SALES
14	JAMES	CLERK	SALES
15	NULL	NULL	OPERATIONS

# JOINTURE EXTERNE FULL OUTER JOIN

FULL JOIN = INNER JOIN + tout enregistrement

supplémentaire dans la table de droite + tout

enregistrement supplémentaire dans la table de gauche

```
SELECT ename , job , dname FROM emp e  
FULL JOIN dept d ON e.dept = d.deptno
```

	ename	job	dname
1	SMITH	CLERK	RESEARCH
2	ALLEN	SALESMAN	SALES
3	WARD	SALESMAN	SALES
4	JONES	MANAGER	RESEARCH
5	MARTIN	SALESMAN	SALES
6	BLAKE	MANAGER	SALES
7	CLARK	MANAGER	ACCOUNTING
8	SCOTT	ANALYST	RESEARCH
9	KING	PRESIDENT	ACCOUNTING
10	TURNER	SALESMAN	SALES
11	ADAMS	CLERK	RESEARCH
12	JAMES	CLERK	SALES
13	FORD	ANALYST	RESEARCH
14	MILLER	CLERK	ACCOUNTING
15	MOHAMED	PRESIDENT	NULL
16	DRISS	MANAGER	NULL
17	HICHAM	ANALYST	NULL
18	IMAD	SALESMAN	NULL
19	AHMED	CLERK	NULL
20	NULL	NULL	OPERATIONS

# RÉSUMÉ

Join kind	Icon	Description
Left outer		All rows from the left table, matching rows from the right table
Right outer		All rows from the right table, matching rows from the left table
Full outer		All rows from both tables
Inner		Only matching rows from both tables

# **REQUÊTES IMBRIQUÉES (SOUS-REQUÊTE), DÉFINITION**

- **Une sous requête** est une requête avec la commande SELECT imbriquée avec les autres commandes (SELECT, UPDATE, INSERT DELETE et CREATE)
- Une sous-requête, peut être utilisée dans les clauses suivantes :
  - **La clause WHERE** d'une instruction UPDATE, DELETE et SELECT
  - **La clause FROM** de l'instruction SELECT
  - **La clause VALUES** de l'instruction INSERT INTO
  - **La clause SET** de l'instruction UPDATE
  - **La création du table**
- On utilise les sous-requête lorsque les jointures ne sont pas possibles

# **SOUS-REQUÊTES : DANS LA CLAUSE WHERE CAS, D'UNE REQUÊTE SELECT**

- Ce type de sous-requête permet de comparer une valeur de la clause WHERE avec le résultat retourné par une sous-requête, dans ce cas on utilise les opérateurs de comparaison suivant :  
 $=$  ,  $!=$  ,  $<$  ,  $\leq$  ,  $>$  ,  $\geq$  , IN.
- L'écriture d'une telle requête pourrait se présenter sous la syntaxe suivante

```
SELECT colonne1, colonne2, colonnex ....
FROM nom_tableA
WHERE colonnex =
    (
        SELECT colonnex from nom_tableB....
        .. Suite requête
    )
Suite requête
```

# SOUS-REQUÊTES : DANS LA CLAUSE WHERE CAS, D'UNE REQUÊTE SELECT

1. La requête en vert est appelée sous-requête
2. Une sous-requête est obligatoirement entre parenthèses. Il ne doit pas comporter de clause ORDER BY, mais peut inclure la clause GROUP BY et HAVING
3. L'opérateur = pourrait être remplacé par un des opérateurs de l'acette précéente
4. La colonne du WHERE de la première requête est obligatoirement la colonne du SELECT de la sous-requête. Une fonction de groupement pourrait être appliquée à la colonnex de la sousrequête
5. La première requête et la sous-requête pourraient porter sur la même table. Dans ce cas nom\_tableA est identique à nom\_tableB
6. Dans le premier SELECT, colonnex n'est pas obligée d'être dans le SELECT

```
SELECT colonne1,colonne2,colonnex...
FROM nom_tableA
WHERE colonnex =
(
    SELECT colonnex from nom_tableB...
    .. Suite requête
)
Suite requête
```

**Requête 1 :** La question: On cherche le nom des employés qui travaillent dans le même département que l'employé FORD

Étape 1: on cherche le département (deptno) de l'employé FORD

Étape 2: on cherche le ename des employés ayant le même deptno que FORD

```
SELECT ename FROM emp
WHERE dept =
(
  SELECT dept FROM emp
  WHERE ename = 'FORD'
);
```

	empno	ename	sal	dept
1	7369	SMITH	800.00	20
2	7499	ALLEN	1600.00	30
3	7521	WARD	1250.00	30
4	7566	JONES	2975.00	20
5	7654	MARTIN	1250.00	30
6	7698	BLAKE	2850.00	30
7	7782	CLARK	2450.00	10
8	7788	SCOTT	3000.00	20
9	7839	KING	5000.00	10
10	7844	TURNER	1500.00	30
11	7876	ADAMS	1100.00	20
12	7900	JAMES	950.00	30
13	7902	FORD	3000.00	20
14	7934	MILLER	1300.00	10
15	7935	MOHAMED	5000.00	NULL
16	7940	DRISS	3500.00	NULL
17	7950	HICHAM	5000.00	NULL
18	7951	IMAD	5000.00	NULL
19	7952	AHMED	5000.00	NULL

	ename
1	SMITH
2	JONES
3	SCOTT
4	ADAMS
5	FORD

**Requête 2 :** On cherche le nom de l'employé qui a le plus haut salaire. :MAX(sal)

L'idée est d'aller chercher le plus haut salaire (MAX), puis chercher le nom de l'employé à qui ça correspond.

Étape 1: on cherche le salaire le plus élevé:

Étape 2: on cherche le ename avec le plus haut salaire

	empno	ename	sal	dept
1	7369	SMITH	800.00	20
2	7499	ALLEN	1600.00	30
3	7521	WARD	1250.00	30
4	7566	JONES	2975.00	20
5	7654	MARTIN	1250.00	30
6	7698	BLAKE	2850.00	30
7	7782	CLARK	2450.00	10
8	7788	SCOTT	3000.00	20
9	7839	KING	5000.00	10
10	7844	TURNER	1500.00	30
11	7876	ADAMS	1100.00	20
12	7900	JAMES	950.00	30
13	7902	FORD	3000.00	20
14	7934	MILLER	1300.00	10
15	7935	MOHAMED	5000.00	NULL
16	7940	DRISS	3500.00	NULL
17	7950	HICHAM	5000.00	NULL
18	7951	IMAD	5000.00	NULL
19	7952	AHMED	5000.00	NULL

```
SELECT ename FROM emp  
WHERE sal =  
(  
SELECT max(sal) FROM emp  
);
```

	ename
1	KING
2	MOHAMED
3	HICHAM
4	IMAD
5	AHMED

**Requête 3 :** La question: On cherche le nom, ename des employés du département SALES.

Étape 1: On cherche le deptno du département SALES

Étape 2, On cherche les employés ayant le deptno

**Remarque:** la requête précédente aurait pu s'écrire avec une jointure.

**Attention:** Il faut toujours utiliser des jointures à la place de sous requête. C'est une obligation

```
SELECT ename FROM emp
WHERE dept =
(
  SELECT deptno FROM dept
  WHERE dname = 'SALES'
);
```

	empno	ename	sal	dept
1	7369	SMITH	800.00	20
2	7499	ALLEN	1600.00	30
3	7521	WARD	1250.00	30
4	7566	JONES	2975.00	20
5	7654	MARTIN	1250.00	30
6	7698	BLAKE	2850.00	30
7	7782	CLARK	2450.00	10
8	7788	SCOTT	3000.00	20
9	7839	KING	5000.00	10
10	7844	TURNER	1500.00	30
11	7876	ADAMS	1100.00	20
12	7900	JAMES	950.00	30
13	7902	FORD	3000.00	20
14	7934	MILLER	1300.00	10
15	7935	MOHAMED	5000.00	NULL
16	7940	DRISS	3500.00	NULL
17	7950	HICHAM	5000.00	NULL
18	7951	IMAD	5000.00	NULL
19	7952	AHMED	5000.00	NULL

	deptno	dname	loc
1	10	ACCOUNTING	NEW YORK
2	20	RESEARCH	DALLAS
3	30	SALES	CHICAGO
4	40	OPERATIONS	BOSTON

	ename
1	ALLEN
2	WARD
3	MARTIN
4	BLAKE
5	TURNER
6	JAMES

**Requête 4:** On cherche le nom des employés qui travaillent dans le même département que l'employé FORD ou l'employé MILLER

La sous-requête renvoie plus qu'une valeur, elle renvoie le deptno 20 et le deptno 10. Dans ce cas il faut utiliser IN à la place de =

Attention: La requête interne renvoie plus qu'un résultat. Il faut utiliser IN dans ce cas.

Si = est utilisé dans la requête alors vous aurez cette erreur

```
SELECT ename FROM emp
WHERE dept IN
(
  SELECT dept FROM emp
  WHERE ename = 'FORD' OR ename
  = 'MILLER'
);
```

	empno	ename	sal	dept
1	7369	SMITH	800.00	20
2	7499	ALLEN	1600.00	30
3	7521	WARD	1250.00	30
4	7566	JONES	2975.00	20
5	7654	MARTIN	1250.00	30
6	7698	BLAKE	2850.00	30
7	7782	CLARK	2450.00	10
8	7788	SCOTT	3000.00	20
9	7839	KING	5000.00	10
10	7844	TURNER	1500.00	30
11	7876	ADAMS	1100.00	20
12	7900	JAMES	950.00	30
13	7902	FORD	3000.00	20
14	7934	MILLER	1300.00	10
15	7935	MOHAMED	5000.00	NULL
16	7940	DRISS	3500.00	NULL
17	7950	HICHAM	5000.00	NULL
18	7951	IMAD	5000.00	NULL
19	7952	AHMED	5000.00	NULL

	deptno	dname	loc
1	10	ACCOUNTING	NEW YORK
2	20	RESEARCH	DALLAS
3	30	SALES	CHICAGO
4	40	OPERATIONS	BOSTON

	ename
1	SMITH
2	JONES
3	CLARK
4	SCOTT
5	KING
6	ADAMS
7	FORD
8	MILLER

# **SOUS-REQUÊTES : DANS LA CLAUSE **WHERE** CAS, D'UNE REQUÊTE SELECT**

## **Les opérateurs ANY et ALL**

Ces deux opérateurs s'utilisent lorsque la sous requête retourne plus qu'un enregistrement.

- On utilise l'opérateur **ANY** pour que la comparaison se fasse pour toutes les valeurs retournées. Le résultat est vrai si au moins une des valeurs répond à la comparaison
- On utilise l'opérateur **ALL** pour que la comparaison se fasse pour toutes les valeurs retournées. Le résultat est vrai si toutes les valeurs répondent à la comparaison

**Requête 5:** Nous souhaitons chercher le nom des employés du département numéro 10 dont le salaire est plus grand que TOUS les employés du département 30

	empno	ename	sal	dept
1	7369	SMITH	800.00	20
2	7499	ALLEN	1600.00	30
3	7521	WARD	1250.00	30
4	7566	JONES	2975.00	20
5	7654	MARTIN	1250.00	30
6	7698	BLAKE	2850.00	30
7	7782	CLARK	2450.00	10
8	7788	SCOTT	3000.00	20
9	7839	KING	5000.00	10
10	7844	TURNER	1500.00	30
11	7876	ADAMS	1100.00	20
12	7900	JAMES	950.00	30
13	7902	FORD	3000.00	20
14	7934	MILLER	1300.00	10
15	7935	MOHAMED	5000.00	NULL
16	7940	DRISS	3500.00	NULL
17	7950	HICHAM	5000.00	NULL
18	7951	IMAD	5000.00	NULL
19	7952	AHMED	5000.00	NULL

```
SELECT ename FROM emp
WHERE dept =10 AND sal > ALL
(
SELECT sal FROM emp
WHERE dept=30
);
```

ename
KING

**Requête 6:** Nous souhaitons chercher le nom des employés du département numéro 10 dont le salaire est plus grand que n'importe lequel des salaires du département 30

	empno	ename	sal	dept
1	7369	SMITH	800.00	20
2	7499	ALLEN	1600.00	30
3	7521	WARD	1250.00	30
4	7566	JONES	2975.00	20
5	7654	MARTIN	1250.00	30
6	7698	BLAKE	2850.00	30
7	7782	CLARK	2450.00	10
8	7788	SCOTT	3000.00	20
9	7839	KING	5000.00	10
10	7844	TURNER	1500.00	30
11	7876	ADAMS	1100.00	20
12	7900	JAMES	950.00	30
13	7902	FORD	3000.00	20
14	7934	MILLER	1300.00	10
15	7935	MOHAMED	5000.00	NULL
16	7940	DRISS	3500.00	NULL
17	7950	HICHAM	5000.00	NULL
18	7951	IMAD	5000.00	NULL
19	7952	AHMED	5000.00	NULL

```
SELECT ename FROM emp
WHERE dept =10 AND sal > ANY
(
SELECT sal FROM emp
WHERE dept=30
);
```

	ename
1	CLARK
2	KING
3	MILLER

# SOUS-REQUÊTES : REQUÊTE SELECT

Supposons que nous voulions trouver la valeur moyenne des commandes pour chacun des clients avec le nom du client.

```
SELECT c.cust_name,  
(  
    SELECT avg(order_value) FROM orders o  
    WHERE o.cust_id = c.cust_id  
) as av_order_value  
FROM customers c;
```

	cust_name	av_order_value
1	Herbert Jones	3396.333333
2	Nancy Powel	3833.333333
3	Daniel Mattis	5200.000000

cust_id	cust_name	cust_city	cust_contact
101	Herbert Jones	Delaware	16633775159
102	Nancy Powell	Philadelphia	16277212992
103	Daniel Mattis	Delaware	16107575525

Table: customers

order_id	cust_id	order_date	order_value
210	103	6/11/2019	5500
211	103	6/11/2019	5100
212	101	6/13/2019	3139
213	102	6/13/2019	3000
214	101	6/13/2019	3550
215	101	6/13/2019	3500
216	103	6/22/2019	5000
217	102	6/22/2019	5500
218	102	6/22/2019	3000

Table: orders

# SOUS-REQUÊTES : DANS LA CLAUSE **FROM** CAS, D'UNE REQUÊTE SELECT

Supposons que nous voulions connaître la valeur totale des commandes passées par les différents clients.

```
SELECT c.cust_id, c.cust_name, o.tot_val
FROM (SELECT cust_id, sum(order_value) tot_val
      FROM orders GROUP BY cust_id) o
INNER JOIN customers c ON o.cust_id=c.cust_id;
```

OR USE THIS

```
SELECT c.cust_id, c.cust_name,
       sum(order_value)as tot_val
  FROM customers c JOIN orders o
    ON o.cust_id=c.cust_id
 GROUP BY c.cust_id,c.cust_name
```

	cust_id	cust_name	tot_val
1	101	Herbert Jones	10189.00
2	102	Nancy Powel	11500.00
3	103	Daniel Mattis	15600.00

cust_id	cust_name	cust_city	cust_contact
101	Herbert Jones	Delaware	16633775159
102	Nancy Powell	Philadelphia	16277212992
103	Daniel Mattis	Delaware	16107575525

Table: customers

order_id	cust_id	order_date	order_value
210	103	6/11/2019	5500
211	103	6/11/2019	5100
212	101	6/13/2019	3139
213	102	6/13/2019	3000
214	101	6/13/2019	3550
215	101	6/13/2019	3500
216	103	6/22/2019	5000
217	102	6/22/2019	5500
218	102	6/22/2019	3000

Table: orders

# SOUS-REQUÊTES : DANS LA CLAUSE SET D'UNE REQUÊTE UPDATE

On peut chercher des valeurs existantes dans la base de données afin de mettre à jour des données d'une table.

La sous-requête est une requête SELECT qui ne doit ramener qu'un seul résultat.

**Requête :** Nous souhaitons mettre à jour les commissions null, par la moyenne de toutes les commissions.

Les commissions (comm) a NULL seront mises à jour par la valeur: 480 qui est la moyennes de toutes les comm

```
UPDATE emp SET comm =  
(  
    SELECT AVG(comm) FROM emp  
)  
WHERE comm IS NULL
```

	ename	sal	comm
1	SMITH	800.00	NULL
2	ALLEN	1600.00	300.00
3	WARD	1250.00	500.00
4	JONES	2975.00	NULL
5	MARTIN	1250.00	1400.00
6	BLAKE	2850.00	NULL
7	CLARK	2450.00	NULL
8	SCOTT	3000.00	NULL
9	KING	5000.00	NULL
10	TURNER	1500.00	0.00
11	ADAMS	1100.00	NULL
12	JAMES	950.00	NULL
13	FORD	3000.00	NULL
14	MILLER	1300.00	NULL
15	MOHAMED	5000.00	NULL
16	DRISS	3500.00	NULL
17	HICHAM	5000.00	NULL
18	IMAD	5000.00	200.00
19	AHMED	5000.00	NULL

	ename	sal	comm
1	SMITH	800.00	480.00
2	ALLEN	1600.00	300.00
3	WARD	1250.00	500.00
4	JONES	2975.00	480.00
5	MARTIN	1250.00	1400.00
6	BLAKE	2850.00	480.00
7	CLARK	2450.00	480.00
8	SCOTT	3000.00	480.00
9	KING	5000.00	480.00
10	TURNER	1500.00	0.00
11	ADAMS	1100.00	480.00
12	JAMES	950.00	480.00
13	FORD	3000.00	480.00
14	MILLER	1300.00	480.00
15	MOHAMED	5000.00	480.00
16	DRISS	3500.00	480.00
17	HICHAM	5000.00	480.00
18	IMAD	5000.00	200.00
19	AHMED	5000.00	480.00

# SOUS-REQUÊTES : DANS LA CLAUSE **VALUES** D'UNE REQUÊTE **INSERT INTO**

Ce type de sous-requête permet d'insérer des données dans une table à partir d'une autre table. **La sous requête** est utilisée **à la place** de la clause **VALUES** de la requête principale et peut retourner plusieurs résultats.

```
insert into etudiant (numetd,nametd) (
SELECT empno,ename from emp WHERE dept = 10
);
```

	ename	sal	comm
1	SMITH	800.00	NULL
2	ALLEN	1600.00	300.00
3	WARD	1250.00	500.00
4	JONES	2975.00	NULL
5	MARTIN	1250.00	1400.00
6	BLAKE	2850.00	NULL
7	CLARK	2450.00	NULL
8	SCOTT	3000.00	NULL
9	KING	5000.00	NULL
10	TURNER	1500.00	0.00
11	ADAMS	1100.00	NULL
12	JAMES	950.00	NULL
13	FORD	3000.00	NULL
14	MILLER	1300.00	NULL
15	MOHAMED	5000.00	NULL
16	DRISS	3500.00	NULL
17	HICHAM	5000.00	NULL
18	IMAD	5000.00	200.00
19	AHMED	5000.00	NULL

	numetd	name
1	7782	CLARK
2	7839	KING
3	7934	MILLER

# CRÉATION DU TABLE PAR SELECT

Ce type de requête est utilisé pour créer une table à partir d'une ou plusieurs table existantes.

La table créée contient les données issues d'une ou de plusieurs table.

Pour créer une table avec une sous requête, on utilise **la syntaxe:**

select <column list> into <table name> from <source> where <whereclause>

```
SELECT * INTO manager FROM emp WHERE job = 'MANAGER'
```