

**Afficher des données provenant de plusieurs
tables à l'aide de jointures**

Obtenir des données à partir de plusieurs tables

EMPLOYEES

	EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
1	200	Whalen	10
2	201	Hartstein	20
3	202	Fay	20
...			
18	174	Abel	80
19	176	Taylor	80
20	178	Grant	(null)

DEPARTMENTS

	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
1	10	Administration	1700
2	20	Marketing	1800
3	50	Shipping	1500
4	60	IT	1400
5	80	Sales	2500
6	90	Executive	1700
7	110	Accounting	1700
8	190	Contracting	1700



	EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
1	200	10	Administration
2	201	20	Marketing
3	202	20	Marketing
4	124	50	Shipping
...			
18	205	110	Accounting
19	206	110	Accounting

Types de jointure

Jointures compatibles avec la norme SQL:1999 :

- Jointures naturelles :
 - Clause NATURAL JOIN
 - Clause USING
 - Clause ON
- Jointure externe (OUTER) :
 - LEFT OUTER JOIN
 - RIGHT OUTER JOIN
 - FULL OUTER JOIN
- Jointures croisées

Joindre des tables à l'aide de la syntaxe SQL:1999

Utilisez une jointure pour effectuer une interrogation de données à partir de plusieurs tables :

```
SELECT table1.column, table2.column
FROM   table1
[NATURAL JOIN table2] |
[JOIN table2 USING (column_name)] |
[JOIN table2
  ON (table1.column_name = table2.column_name)] |
[LEFT|RIGHT|FULL OUTER JOIN table2
  ON (table1.column_name = table2.column_name)] |
[CROSS JOIN table2];
```

Qualifier les noms de colonne ambigus

- Utilisez des préfixes de table pour qualifier le nom des colonnes qui se trouvent dans plusieurs tables.
- Utilisez des préfixes de table pour améliorer les performances.
- Au lieu d'utiliser des préfixes pour définir des noms de table complets, vous pouvez utiliser des alias de table.
- Les alias de table permettent d'attribuer des noms plus courts aux tables :
 - Code SQL plus compact, utilisation de moins de mémoire
- Utilisez des alias de colonne pour distinguer les colonnes portant des noms identiques mais résidant dans des tables différentes.

Créer des jointures naturelles

- La clause `NATURAL JOIN` est basée sur toutes les colonnes des deux tables qui portent le même nom.
- Elle sélectionne les lignes des deux tables qui présentent des valeurs identiques dans les colonnes de même nom.
- Si les colonnes portant le même nom présentent des types de données différents, une erreur est renvoyée.

Extraire des enregistrements avec des jointures naturelles

```
SELECT department_id, department_name,  
       location_id, city  
FROM   departments  
NATURAL JOIN locations ;
```

	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID	CITY
1	60	IT	1400	Southlake
2	50	Shipping	1500	South San Francisco
3	10	Administration	1700	Seattle
4	90	Executive	1700	Seattle
5	110	Accounting	1700	Seattle
6	190	Contracting	1700	Seattle
7	20	Marketing	1800	Toronto
8	80	Sales	2500	Oxford

Créer des jointures à l'aide de la clause USING

- Si plusieurs colonnes portent le même nom mais présentent des types de données différents, utilisez la clause `USING` pour indiquer les colonnes pour une équijointure.
- Indiquez la clause `USING` pour mettre en correspondance une seule colonne lorsque la correspondance concerne plusieurs colonnes.
- Les clauses `NATURAL JOIN` et `USING` sont mutuellement exclusives.

Joindre des noms de colonne



EMPLOYEES

	 EMPLOYEE_ID	 DEPARTMENT_ID
1	200	10
2	201	20
3	202	20
4	205	110
5	206	110
6	100	90
7	101	90
8	102	90
9	103	60
10	104	60

...

**Clé
étrangère**

DEPARTMENTS

	 DEPARTMENT_ID	 DEPARTMENT_NAME
1	10	Administration
2	20	Marketing
3	50	Shipping
4	60	IT
5	80	Sales
6	90	Executive
7	110	Accounting
8	190	Contracting

**Clé
primaire**

Extraire des enregistrements avec la clause USING

```
SELECT employee_id, last_name,  
       location_id, department_id  
FROM   employees JOIN departments  
USING (department_id) ;
```

	<small>A Z</small> EMPLOYEE_ID	<small>A Z</small> LAST_NAME	<small>A Z</small> LOCATION_ID	<small>A Z</small> DEPARTMENT_ID
1	200	Whalen	1700	10
2	201	Hartstein	1800	20
3	202	Fay	1800	20
4	144	Vargas	1500	50
5	143	Matos	1500	50
6	142	Davies	1500	50
7	141	Rajs	1500	50
8	124	Mourgos	1500	50

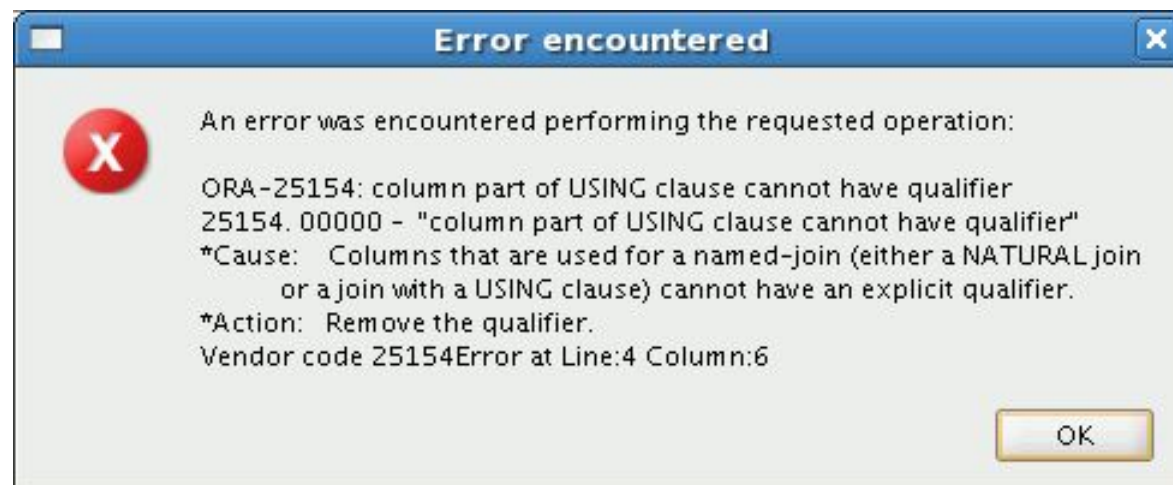
...

18	206	Gietz	1700	110
19	205	Higgins	1700	110

Utiliser des alias de table avec la clause USING

- Ne qualifiez pas une colonne utilisée dans la clause USING.
- Si la même colonne est utilisée ailleurs dans l'instruction SQL, ne lui attribuez pas d'alias.

```
SELECT l.city, d.department_name  
FROM   locations l JOIN departments d  
USING (location_id)  
WHERE  d.location_id = 1400;
```



Créer des jointures à l'aide de la clause ON

- La condition de jointure d'une jointure naturelle est en fait une équijointure de toutes les colonnes portant le même nom.
- Utilisez la clause `ON` pour indiquer des conditions arbitraires ou les colonnes à joindre.
- La condition de jointure est séparée des autres conditions de recherche.
- La clause `ON` facilite la compréhension du code.

Extraire des enregistrements avec la clause ON

```
SELECT e.employee_id, e.last_name, e.department_id,  
       d.department_id, d.location_id  
FROM   employees e JOIN departments d  
ON     (e.department_id = d.department_id);
```

	EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_ID_1	LOCATION_ID
1	200	Whalen	10	10	1700
2	201	Hartstein	20	20	1800
3	202	Fay	20	20	1800
4	144	Vargas	50	50	1500
5	143	Matos	50	50	1500
6	142	Davies	50	50	1500
7	141	Rajs	50	50	1500
8	124	Mourgos	50	50	1500
9	103	Hunold	60	60	1400
10	104	Ernst	60	60	1400
11	107	Lorentz	60	60	1400

...

Créer des jointures à trois liens à l'aide de la clause ON

```
SELECT employee_id, city, department_name
FROM   employees e
JOIN   departments d
ON     d.department_id = e.department_id
JOIN   locations l
ON     d.location_id = l.location_id;
```

	 EMPLOYEE_ID	 CITY	 DEPARTMENT_NAME
1	100	Seattle	Executive
2	101	Seattle	Executive
3	102	Seattle	Executive
4	103	Southlake	IT
5	104	Southlake	IT
6	107	Southlake	IT
7	124	South San Francisco	Shipping
8	141	South San Francisco	Shipping
9	142	South San Francisco	Shipping

...

Appliquer des conditions supplémentaires à une jointure

Utilisez la clause `AND` ou la clause `WHERE` pour appliquer des conditions supplémentaires :

```
SELECT e.employee_id, e.last_name, e.department_id,  
       d.department_id, d.location_id  
FROM   employees e JOIN departments d  
ON     (e.department_id = d.department_id)  
AND    e.manager_id = 149 ;
```

O

u

```
SELECT e.employee_id, e.last_name, e.department_id,  
       d.department_id, d.location_id  
FROM   employees e JOIN departments d  
ON     (e.department_id = d.department_id)  
WHERE  e.manager_id = 149 ;
```

Joindre une table à elle-même

EMPLOYEES (WORKER)

EMPLOYEE_ID	LAST_NAME	MANAGER_ID
200	Whalen	101
201	Hartstein	100
202	Fay	201
205	Higgins	101
206	Gietz	205
100	King	(null)
101	Kochhar	100
102	De Haan	100
103	Hunold	102
104	Ernst	103

...

EMPLOYEES (MANAGER)

EMPLOYEE_ID	LAST_NAME
200	Whalen
201	Hartstein
202	Fay
205	Higgins
206	Gietz
100	King
101	Kochhar
102	De Haan
103	Hunold
104	Ernst

...



La colonne `MANAGER_ID` de la table `WORKER` est égale à la colonne `EMPLOYEE_ID` de la table `MANAGER`.

Auto-jointures à l'aide de la clause ON

```
SELECT worker.last_name emp, manager.last_name mgr
FROM   employees worker JOIN employees manager
ON     (worker.manager_id = manager.employee_id);
```

	EMP	MGR
1	Hunold	De Haan
2	Fay	Hartstein
3	Gietz	Higgins
4	Lorentz	Hunold
5	Ernst	Hunold
6	Zlotkey	King
7	Mourgos	King

...

Non-équijointures

EMPLOYEES

	<small>AZ</small> LAST_NAME	<small>AZ</small> SALARY
1	Whalen	4400
2	Hartstein	13000
3	Fay	6000
4	Higgins	12000
5	Gietz	8300
6	King	24000
7	Kochhar	17000
8	De Haan	17000
9	Hunold	9000
10	Ernst	6000
...		
19	Taylor	8600
20	Grant	7000

JOB_GRADES

	<small>AZ</small> GRADE_LEVEL	<small>AZ</small> LOWEST_SAL	<small>AZ</small> HIGHEST_SAL
1	A	1000	2999
2	B	3000	5999
3	C	6000	9999
4	D	10000	14999
5	E	15000	24999
6	F	25000	40000

La table **JOB_GRADES** définit la plage de valeurs **LOWEST_SAL** et **HIGHEST_SAL** pour chaque valeur **GRADE_LEVEL**. Par conséquent, la colonne **GRADE_LEVEL** peut être utilisée pour affecter un niveau à chaque employé.

Extraire des enregistrements à l'aide de non-équijointures

```
SELECT e.last_name, e.salary, j.grade_level
FROM   employees e JOIN job grades j
ON     e.salary
      BETWEEN j.lowest_sal AND j.highest_sal;
```

	LAST_NAME	SALARY	GRADE_LEVEL
1	Vargas	2500	A
2	Matos	2600	A
3	Davies	3100	B
4	Rajs	3500	B
5	Lorentz	4200	B
6	Whalen	4400	B
7	Mourgos	5800	B
8	Ernst	6000	C
9	Fay	6000	C
10	Grant	7000	C

...

Renvoyer des enregistrements sans correspondance directe à l'aide de jointures externes

DEPARTMENTS

	DEPARTMENT_NAME	DEPARTMENT_ID
1	Administration	10
2	Marketing	20
3	Shipping	50
4	IT	60
5	Sales	80
6	Executive	90
7	Accounting	110
8	Contracting	190

Le département 190 ne comporte aucun employé.

L'employé "Grant" n'a été associé à un ID de département.

Équijointure avec EMPLOYEES

	DEPARTMENT_ID	LAST_NAME
1	10	Whalen
2	20	Hartstein
3	20	Fay
4	110	Higgins
5	110	Gietz
6	90	King
7	90	Kochhar
8	90	De Haan
9	60	Hunold
10	60	Ernst

...

18	80	Abel
19	80	Taylor

Comparaison entre les jointures INNER et OUTER

- Dans la syntaxe SQL:1999, la jointure de deux tables qui ne renvoie que les lignes qui correspondent est appelée jointure interne.
- Une jointure entre deux tables qui renvoie les résultats de la jointure interne, ainsi que les lignes sans correspondance de la table de gauche (ou de droite) est appelée jointure externe gauche (ou droite).
- Une jointure entre deux tables qui renvoie les résultats d'une jointure interne, ainsi que les résultats d'une jointure droite et d'une jointure gauche est une jointure externe complète.

LEFT OUTER JOIN

```
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e LEFT OUTER JOIN departments d
ON     (e.department_id = d.department_id) ;
```

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Fay	20	Marketing
3	Hartstein	20	Marketing
4	Vargas	50	Shipping
5	Matos	50	Shipping

...

16	Kochhar	90	Executive
17	King	90	Executive
18	Gietz	110	Accounting
19	Higgins	110	Accounting
20	Grant	(null)	(null)

RIGHT OUTER JOIN

```
SELECT e.last_name, d.department_id, d.department_name
FROM   employees e RIGHT OUTER JOIN departments d
ON     (e.department_id = d.department_id) ;
```

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Hartstein	20	Marketing
3	Fay	20	Marketing
4	Davies	50	Shipping
5	Vargas	50	Shipping
6	Rajs	50	Shipping
7	Mourgos	50	Shipping
8	Matos	50	Shipping

...

18	Higgins	110	Accounting
19	Gietz	110	Accounting
20	(null)	190	Contracting

FULL OUTER JOIN

```
SELECT e.last_name, d.department_id, d.department_name
FROM   employees e FULL OUTER JOIN departments d
ON     (e.department_id = d.department_id) ;
```

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Hartstein	20	Marketing
3	Fay	20	Marketing
4	Higgins	110	Accounting

...

17	Zlotkey	80	Sales
18	Abel	80	Sales
19	Taylor	80	Sales
20	Grant	(null)	(null)
21	(null)	190	Contracting

Produits cartésiens

- Un produit cartésien est généré dans les cas suivants :
 - La condition de jointure est omise.
 - La condition de jointure n'est pas valide.
 - Toutes les lignes de la première table sont jointes à toutes les lignes de la deuxième.
- Incluez toujours une condition de jointure valide si vous souhaitez éviter la formation d'un produit cartésien.

Générer un produit cartésien

EMPLOYEES (20

	EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
1	200	Whalen	10
2	201	Hartstein	20
3	202	Fay	20
4	205	Higgins	110
...			
19	176	Taylor	80
20	178	Grant	(null)

DEPARTMENTS (8

	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
1	10	Administration	1700
2	20	Marketing	1800
3	50	Shipping	1500
4	60	IT	1400
5	80	Sales	2500
6	90	Executive	1700
7	110	Accounting	1700
8	190	Contracting	1700



Produit cartésien :
20 x 8 = 160 lignes

	EMPLOYEE_ID	DEPARTMENT_ID	LOCATION_ID
1	200	10	1700
2	201	20	1700
...			
21	200	10	1800
22	201	20	1800
...			
159	176	80	1700
160	178	(null)	1700

Créer des jointures croisées

- La clause `CROSS JOIN` effectue une jointure croisée entre deux tables.
- Cela équivaut au produit cartésien des deux tables.

```
SELECT last_name, department_name  
FROM employees  
CROSS JOIN departments ;
```

	LAST_NAME	DEPARTMENT_NAME
1	Abel	Administration
2	Davies	Administration
3	De Haan	Administration
4	Ernst	Administration
5	Fay	Administration

...

158	Vargas	Contracting
159	Whalen	Contracting
160	Zlotkey	Contracting