



Visualización de Datos de Varias Tablas

ORACLE®
DATABASE **11^g**

Objetivos

Al finalizar esta lección, deberá estar capacitado para:

- **Escribir sentencias SELECT para acceder a datos de más de una tabla utilizando uniones de igualdad**
- **Visualizar datos que generalmente no cumplen una condición de unión utilizando uniones externas**
- **Unir una tabla consigo misma utilizando una autounión**
- **Generar un producto cartesiano de todos los renglones para dos o mas tablas.**

Obteniendo Datos desde Multiples Tablas

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
105	Austin	60
106	Pataballa	60

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
30	Purchasing	1700
40	Human Resources	2400
50	Shipping	1500
60	IT	1400
70	Public Relations	2700
80	Sales	2500
90	Executive	1700



EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
200	10	Administration
201	20	Marketing
202	20	Marketing
114	30	Purchasing

Tipos de Uniones

- **Uniones Cruzadas**
- **Uniones Naturales**
- **Cláusula USING**
- **Uniones externas completas o de dos lados**
- **Condiciones de unión arbitrarias para uniones externas**

Unión de Tablas Utilizando la Sintaxis SQL:

Utilice la unión para consultar datos de más de una tabla.

```
SELECT    table1.column, table2.column
FROM      table1
[CROSS JOIN table2] |
[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)];
```

Creación de Uniones Naturales

- La cláusula **NATURAL JOIN** se basa en todas las columnas de las dos tablas que tienen el mismo nombre.
- Selecciona filas de las dos tablas que tienen los mismos valores en todas las columnas coincidentes.
- Si las columnas que tienen el mismo nombre tienen distintos tipos de dato, se devuelve un error.

Recuperación de Registros con Uniones Naturales

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

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID	CITY
60	IT	1400	Southlake
50	Shipping	1500	South San Francisco
10	Administration	1700	Seattle
90	Executive	1700	Seattle
30	Purchasing	1700	Seattle
100	Finance	1700	Seattle
120	Treasury	1700	Seattle
140	Control And Credit	1700	Seattle
160	Benefits	1700	Seattle
250	Retail Sales	1700	Seattle
240	Government Sales	1700	Seattle
230	IT Helpdesk	1700	Seattle
220	NOC	1700	Seattle
210	IT Support	1700	Seattle
200	Operations	1700	Seattle

Creación de Uniones con la Cláusula USING

- Si varias columnas tienen los mismos nombres pero los tipos de datos no coinciden, la cláusula **NATURAL JOIN** se puede modificar con a cláusula **USING** para especificar las columnas que se deben utilizar para una unión de igualdad.
- Utilice la cláusula **USING** para hacer coincidir solamente una columna cuando coincidan varias.
- No utilice un nombre o alias de tabla en las columnas de referencia.
- La cláusulas **NATURAL JOIN** y **USING** son mutuamente excluyentes.

Uniendo nombres de columnas

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
105	Austin	60
106	Pataballa	60

Foreign key



DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
30	Purchasing	1700
40	Human Resources	2400
50	Shipping	1500
60	IT	1400
70	Public Relations	2700
80	Sales	2500
90	Executive	1700

Primary key



Recuperación de Registros con la Cláusula USING

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

EMPLOYEE_ID	LAST_NAME	LOCATION_ID
200	Whalen	1700
201	Hartstein	1800
202	Fay	1800
114	Raphaely	1700
115	Khoo	1700
116	Baida	1700
119	Colmenares	1700
118	Himuro	1700
117	Tobias	1700
203	Mavris	2400
120	Weiss	1500
121	Fripp	1500
123	Vollman	1500
132	Olson	1500
131	Marlow	1500
130	Atkinson	1500

Calificación de Nombres de Columna Ambiguas

- **Utilice prefijos de tabla para cualificar nombres de columna que estén en varias tablas.**
- **Mejore el rendimiento utilizando prefijos de tabla.**
- **Distinga las columnas que tengan nombres idénticos pero que residan en tablas diferentes utilizando alias de columna.**
- **No usar alias en columnas que están identificadas en la cláusula USING y enumerado en otra parte en la declaración SQL.**

Uso de Alias de Tabla

- Simplifique las consultas utilizando alias de tabla.
- Mejore el rendimiento utilizando prefijos de tabla.

```
SELECT    [e].employee_id, [e].last_name, [e].department_id,  
          [d].department_id, [d].location_id  
FROM      employees [e], departments [d]  
[UNION DE TABLAS]
```

Creación de Uniones con la Cláusula ON

- La condición de unión para la unión natural es básicamente una unión de igualdad de todas las columnas con el mismo nombre.
- Para especificar condiciones arbitrarias o especificar columnas para unir, se utiliza la cláusula ON.
- La condición de unión se separa de otras condiciones de *búsqueda*.
- La cláusulas ON facilita la comprensión del código.

Recuperación de Registros con la Cláusula 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	LOCATION_ID
200	Whalen	10	10	1700
201	Hartstein	20	20	1800
202	Fay	20	20	1800
114	Raphaely	30	30	1700
115	Khoo	30	30	1700
116	Baida	30	30	1700
119	Colmenares	30	30	1700
118	Himuro	30	30	1700
117	Tobias	30	30	1700
203	Mavris	40	40	2400
120	Weiss	50	50	1500
121	Fripp	50	50	1500
123	Vollman	50	50	1500

Autouniones

EMPLOYEES (WORKER)

EMPLOYEE_ID	LAST_NAME	MANAGER_ID
100	King	
101	Kochhar	100
102	De Haan	100
103	Hunold	102
104	Ernst	103
105	Austin	103
106	Pataballa	103
107	Lorentz	103
108	Greenberg	101
109	Faviet	108
110	Chen	108
111	Sciarra	108
112	Urman	108
113	Popp	108
114	Raphaely	100
115	Khoo	114

EMPLOYEES (MANAGER)

EMPLOYEE_ID	LAST_NAME
100	King
101	Kochhar
102	De Haan
103	Hunold
104	Ernst
105	Austin
106	Pataballa
107	Lorentz
108	Greenberg
109	Faviet
110	Chen
111	Sciarra
112	Urman
113	Popp
114	Raphaely
115	Khoo



MANAGER_ID en la tabla WORKERes igual a
EMPLOYEE_ID en la tabla MANAGER

Self-Joins Usando la cláusula ON

```
SELECT  e.last_name EMP, m.last_name MGR
FROM    employees e JOIN employees m
ON      (e.manager_id = m.employee_id);
```


Condiciones Adicionales

```
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;
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_ID	LOCATION_ID
174	Abel	80	80	2500
175	Hutton	80	80	2500
176	Taylor	80	80	2500
179	Johnson	80	80	2500
177	Livingston	80	80	2500

Creación de Uniones en Tres Sentidos con la Cláusula 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
100	Seattle	Executive
101	Seattle	Executive
102	Seattle	Executive
103	Southlake	IT
104	Southlake	IT
105	Southlake	IT
106	Southlake	IT
107	Southlake	IT
108	Seattle	Finance
109	Seattle	Finance
110	Seattle	Finance
111	Seattle	Finance
112	Seattle	Finance
113	Seattle	Finance

Uniones Externas

DEPARTMENTS

DEPARTMENT_NAME	DEPARTMENT_ID
Administration	10
Marketing	20
Purchasing	30
Human Resources	40
Shipping	50
IT	60
Public Relations	70
Sales	80
Executive	90
Finance	100
Accounting	110
Treasury	120
Corporate Tax	130
Control And Credit	140
Shareholder Services	150
Benefits	160
Manufacturing	170
Construction	180
Contracting	190

EMPLOYEES

DEPARTMENT_ID	LAST_NAME
90	King
90	Kochhar
90	De Haan
60	Hunold
60	Ernst
60	Austin
60	Pataballa
60	Lorentz
100	Greenberg
100	Faviet
100	Chen
100	Sciarra
100	Urman
100	Popp
30	Raphaely
30	Khoo
30	Baida



No hay empleados en el departamento 190.

Uniones INNER frente a OUTER

- La unión de dos tablas que devuelve solamente las filas coincidentes es una unión interna.
- Una unión entre dos tablas que devuelve los resultados de la unión interna así como las tablas izquierda (o derecha) de filas no coincidentes es una unión externa izquierda (o derecha).
- Una unión entre dos tablas que devuelve los resultados de un unión interna así como los de una unión izquierda y derecha es una unión externa completa.

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
Whalen	10	Administration
Fay	20	Marketing
Hartstein	20	Marketing
Colmenares	30	Purchasing
Himuro	30	Purchasing
Tobias	30	Purchasing
Baida	30	Purchasing
Khoo	30	Purchasing
Raphaely	30	Purchasing
Mavris	40	Human Resources
Grant	50	Shipping
OConnell	50	Shipping
Feeney	50	Shipping
Walsh	50	Shipping

RIGHT OUTER JOIN

```
SELECT  e.last_name, e.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
Whalen	10	Administration
Hartstein	20	Marketing
Fay	20	Marketing
Raphaely	30	Purchasing
Khoo	30	Purchasing
Baida	30	Purchasing
Colmenares	30	Purchasing
Himuro	30	Purchasing
Tobias	30	Purchasing
Mavris	40	Human Resources
Weiss	50	Shipping
Fripp	50	Shipping
Vollman	50	Shipping
Olson	50	Shipping

FULL OUTER JOIN

```
SELECT  e.last_name, e.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
Whalen	10	Administration
Fay	20	Marketing
Hartstein	20	Marketing
Colmenares	30	Purchasing
Himuro	30	Purchasing
Tobias	30	Purchasing
Baida	30	Purchasing
Khoo	30	Purchasing
Raphaely	30	Purchasing
Mavris	40	Human Resources
Grant	50	Shipping
OConnell	50	Shipping
Feeney	50	Shipping
Walsh	50	Shipping

Productos Cartesianos

- **Un producto Cartesiano es formado cuando:**
 - Una condición de unión está omitida
 - Una condición de unión no es válida
 - Todas las filas de la primera tabla son unidas a todas las filas de la segunda tabla
- **Para evitar un producto cartesiano, incluya siempre una condición de unión válida en una cláusula WHERE.**

Generación de un Producto Cartesiano

EMPLOYEES (20 filas)

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
105	Austin	60
106	Pataballa	60

DEPARTMENTS (8 filas)

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
30	Purchasing	1700
40	Human Resources	2400
50	Shipping	1500
60	IT	1400
70	Public Relations	2700
80	Sales	2500
90	Executive	1700

Producto cartesiano:

20x8 = 160 filas



EMPLOYEE_ID	DEPARTMENT_ID	LOCATION_ID
200	10	1700
201	20	1800
202	20	1800
114	30	1700
115	30	1700

ORACLE

Creación de Uniones Cruzadas

- La cláusula **CROSS JOIN** produce varios productos entre dos tablas.
- Es lo mismo que un producto Cartesiano entre las dos tablas.

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

LAST_NAME	DEPARTMENT_NAME
King	Administration
Kochhar	Administration
De Haan	Administration
Hunold	Administration
Ernst	Administration
Austin	Administration
Pataballa	Administration
Lorentz	Administration
Greenberg	Administration
Faviet	Administration

Unión de Tablas Utilizando la Sintaxis Oracle

Utilice una unión para consultar datos de más de una tabla :

```
SELECT    table1.column, table2.column  
FROM      table1, table2  
WHERE     table1.column1 = table2.column2;
```

- **Escriba la condición de unión en la cláusula WHERE.**
- **Escriba en el nombre de columna el nombre de tabla como prefijo si aparece el mismo nombre de columna en más de una tabla.**

Que es una unión de Igualdad ?

EMPLOYEES

EMPLOYEE_ID	DEPARTMENT_ID
100	90
101	90
102	90
103	60
104	60
105	60
106	60
107	60
108	100
109	100
110	100
111	100
112	100
113	100
114	30
115	30
116	30
117	30
118	30
119	30
120	50



Clave Ajena

DEPARTMENT

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
30	Purchasing
40	Human Resources
50	Shipping
60	IT
70	Public Relations
80	Sales
90	Executive
100	Finance
110	Accounting
120	Treasury
130	Corporate Tax
140	Control And Credit
150	Shareholder Services
160	Benefits
170	Manufacturing



Clave Primaria

Recuperación de Registros con Uniones de Igualdad

```
SELECT  employees.employee_id, employees.last_name,  
        employees.department_id, departments.department_id,  
        departments.location_id  
FROM    employees, departments  
WHERE   employees.department_id = departments.department_id;
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_ID	LOCATION_ID
200	Whalen	10	10	1700
201	Hartstein	20	20	1800
202	Fay	20	20	1800
114	Raphaely	30	30	1700
115	Khoo	30	30	1700
116	Baida	30	30	1700
119	Colmenares	30	30	1700
118	Himuro	30	30	1700
117	Tobias	30	30	1700
203	Mavris	40	40	2400
120	Weiss	50	50	1500
121	Fripp	50	50	1500

Condiciones de Búsqueda Adicionales utilizando el operador AND

EMPLOYEES

LAST_NAME	DEPARTMENT_ID
Raphaely	30
Baida	30
Kaufing	50
Nayer	50
Landry	50
Markle	50
Marlow	50
Mallin	50
Ladwig	50
Patel	50
Rajs	50
Davies	50
Matos	50
Vargas	50
Taylor	50

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
30	Purchasing
40	Human Resources
50	Shipping
60	IT
70	Public Relations

Unión de Más de Dos Tablas

EMPLOYEES		DEPARTMENTS		LOCATIONS	
LAST_NAME	DEPARTMENT_ID	DEPARTMENT_ID	LOCATION_ID	LOCATION_ID	CITY
King	90	10	1700	1000	Roma
Kochhar	90	20	1800	1100	Venice
De Haan	90	30	1700	1200	Tokyo
Hunold	60	40	2400	1300	Hiroshima
Ernst	60	50	1500	1400	Southlake
Austin	60	60	1400	1500	South San Francisco
Pataballa	60	70	2700		
Lorentz	60	80	2500		
Greenberg	100	90	1700		
Faviet	100	100	1700		
Chen	100	110	1700		
Sciarra	100				
Uman	100				
Popp	100				
Raphaely	30				
Khoo	30				
Baer	30				

- Para unir n tablas, se necesita un mínimo de $n-1$ condiciones de unión. Por ejemplo, para unir tres tablas, se requiere un mínimo de dos uniones.

Unión de una Tabla Consigo Misma

```
SELECT worker.last_name || 'works for'
       || manager.last_name
FROM   employees worker, employees manager
WHERE  worker.manager_id = manager.employee_id;
```

WORKER.LAST_NAME 'TRABAJA PARA' MANAGER.LAST_NAME
Hartstein Trabaja para King
Zlotkey Trabaja para King
Cambrault Trabaja para King
Errazuriz Trabaja para King
Partners Trabaja para King
Russell Trabaja para King
Mourgos Trabaja para King
Vollman Trabaja para King
Kaufling Trabaja para King
Fripp Trabaja para King
Weiss Trabaja para King
Raphaely Trabaja para King
De Haan Trabaja para King
Kochhar Trabaja para King
Higgins Trabaja para Kochhar
Baer Trabaja para Kochhar

Práctica 4: Visión General

Esta práctica cubre los siguientes temas:

- **Unión de Tablas utilizando una unión de igualdad**
- **Realización de uniones externas y auto uniones**
- **Adición de condiciones**