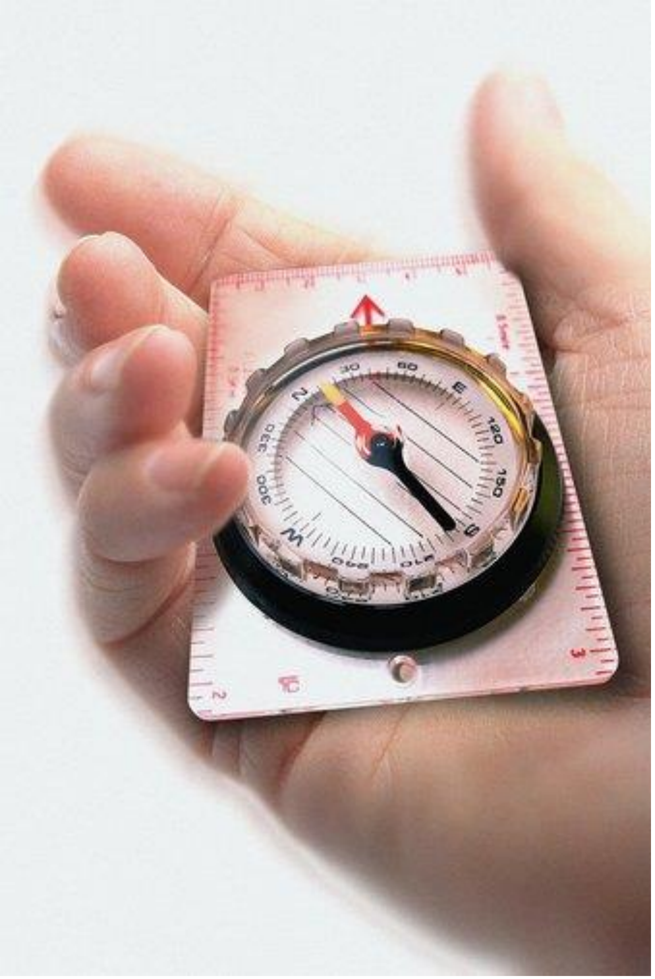
Displaying Data from Multiple Tables

Course objectives

By completing this course, you will be able to:



- Write SELECT statements to access data from more than one table using equijoins and nonequijoins
- Join a table to itself by using a self-join
- View data that generally does not meet a join condition by using outer joins
- Generate a Cartesian product of all rows from two or more tables

Course topics

Course's plan:

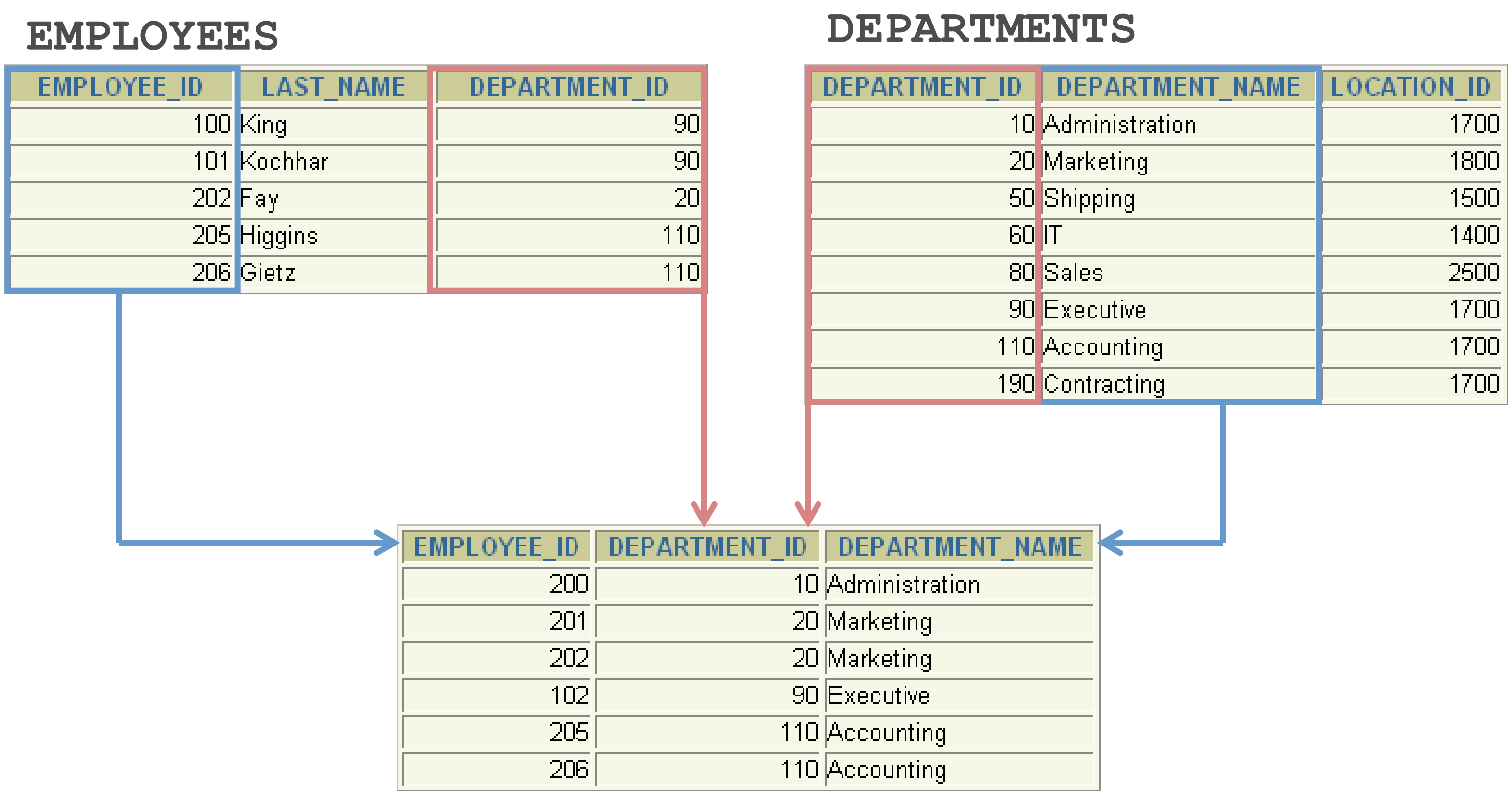
- Displaying Data from Multiple Tables

Displaying Data from Multiple Tables

Preview

- Joins: Presentation.
- Types of joins.
- The **`ON`** Clause.
- Non-equijoins.
- Outer Joins.
- Cross Joins.

Joins: Presentation



EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
202	Fay	20
205	Higgins	110
206	Gietz	110

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700

EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
200	10	Administration
201	20	Marketing
202	20	Marketing
102	90	Executive
205	110	Accounting
206	110	Accounting

Types of Joins

Joining Tables Using SQL:1999 Syntax:

```
 SELECT table1.column, table2.column
 FROM   table1
[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];
```

Qualifying Ambiguous Column Names

- Use table prefixes to qualify column names that are in multiple tables.
- Use table prefixes to improve performance.
- Use column aliases to distinguish columns that have identical names but reside in different tables.
- Do not use aliases on columns that are identified in the **`USING`** clause and listed elsewhere in the SQL statement.

The **ON** Clause

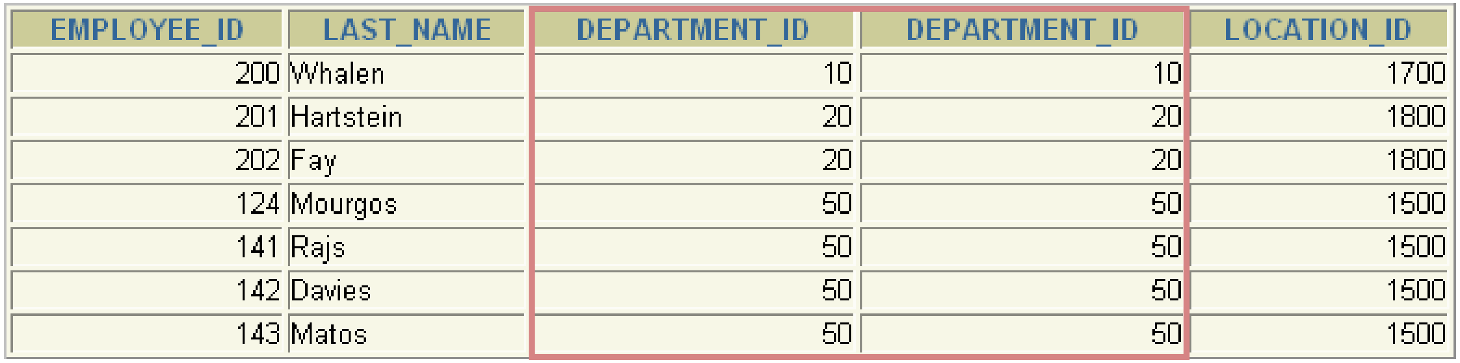
Creating Joins with the **ON** Clause:

- The join condition for the natural join is basically an equijoin of all columns with the same name.
- Use the **ON** clause to specify arbitrary conditions or specify columns to join.
- The join condition is separated from other search conditions.
- The **ON** clause makes code easy to understand.

The ON Clause

Retrieving Records with the ON Clause:

```
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
124	Mourgos	50	50	1500
141	Rajs	50	50	1500
142	Davies	50	50	1500
143	Matos	50	50	1500

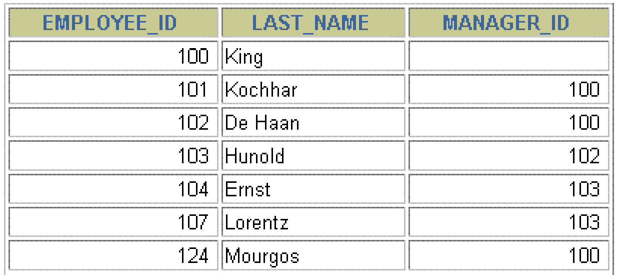
...

19 rows selected.

The ON Clause

Self-Joins Using the ON Clause:

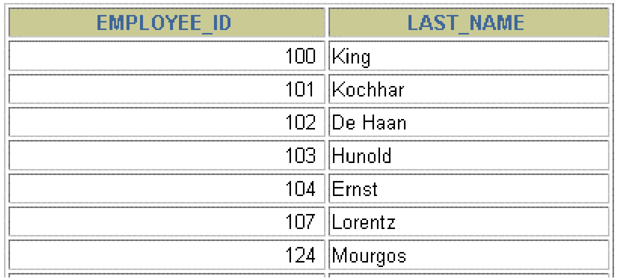
EMPLOYEES (WORKER)



EMPLOYEE_ID	LAST_NAME	MANAGER_ID
100	King	
101	Kochhar	100
102	De Haan	100
103	Hunold	102
104	Ernst	103
107	Lorentz	103
124	Mourgos	100

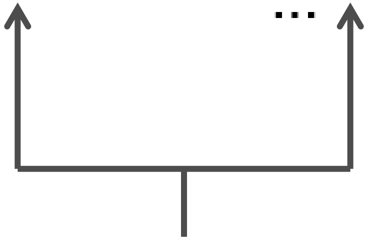
...

EMPLOYEES (MANAGER)



EMPLOYEE_ID	LAST_NAME
100	King
101	Kochhar
102	De Haan
103	Hunold
104	Ernst
107	Lorentz
124	Mourgos

...

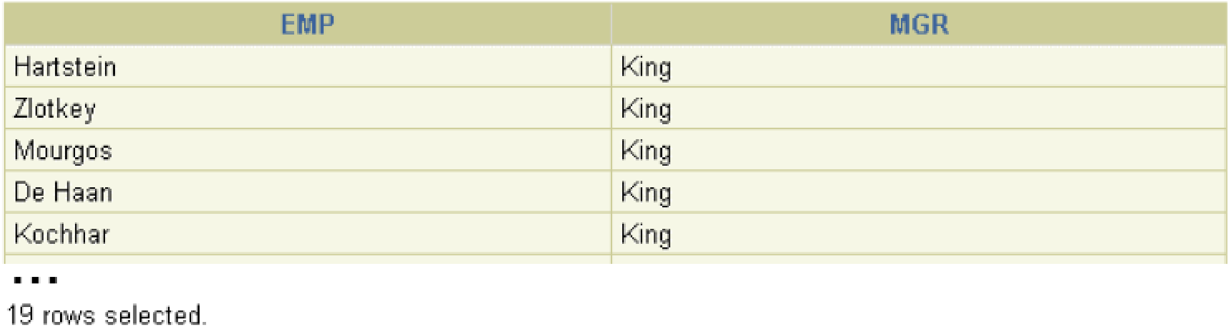


MANAGER_ID in the **WORKER** table is equal to **EMPLOYEE_ID** in the **MANAGER** table.

The ON Clause

Self-Joins Using the ON Clause:

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



EMP	MGR
Hartstein	King
Zlotkey	King
Mourgos	King
De Haan	King
Kochhar	King

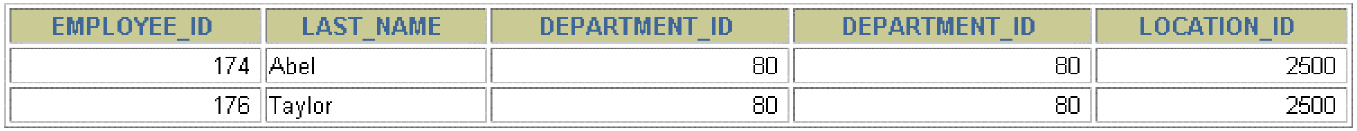
...

19 rows selected.

The ON Clause

Applying Additional Conditions to a Join:

```
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
176	Taylor	80	80	2500

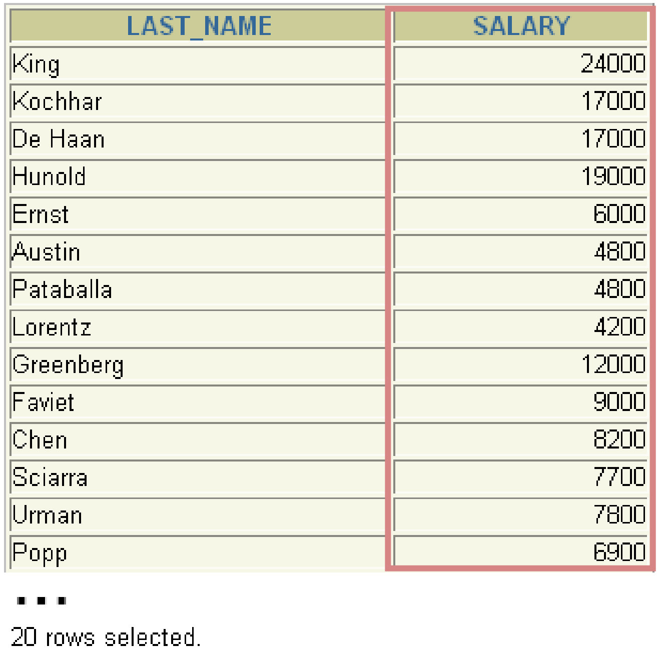
The ON Clause

Creating Three-Way Joins with the ON Clause:

```
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;
```

Non-equijoins

EMPLOYEES

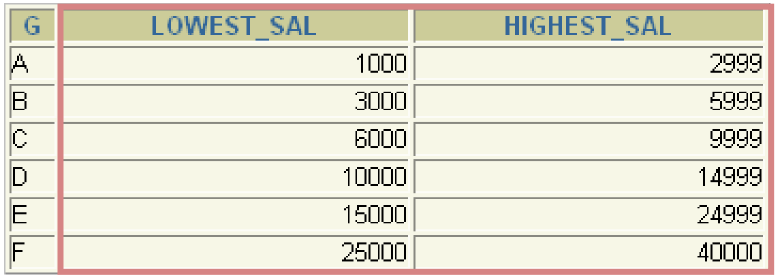


LAST_NAME	SALARY
King	24000
Kochhar	17000
De Haan	17000
Hunold	19000
Ernst	6000
Austin	4800
Pataballa	4800
Lorentz	4200
Greenberg	12000
Faviet	9000
Chen	8200
Sciarra	7700
Urman	7800
Popp	6900

...

20 rows selected.

JOB_GRADES



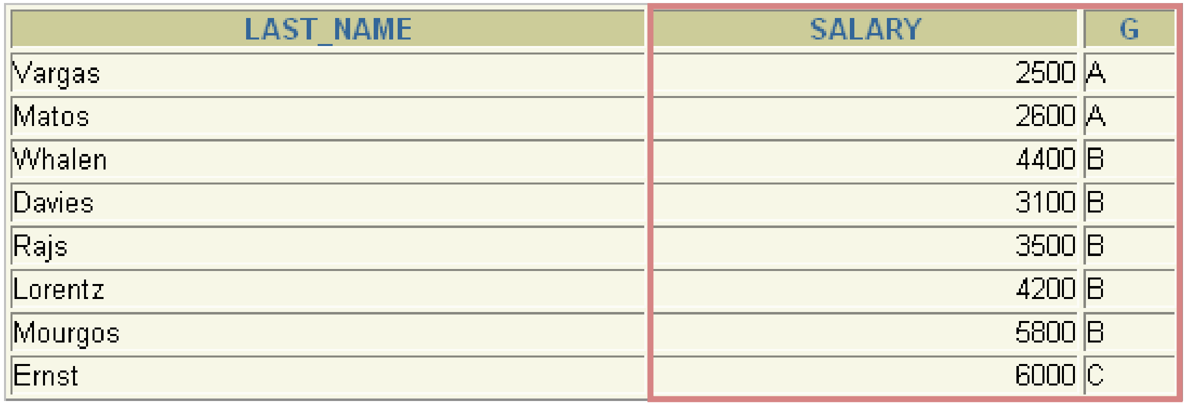
G	LOWEST_SAL	HIGHEST_SAL
A	1000	2999
B	3000	5999
C	6000	9999
D	10000	14999
E	15000	24999
F	25000	40000

Salary in the **EMPLOYEES** table must be between lowest salary and highest salary in the **JOB_GRADES** table.

Non-equijoins

Retrieving Records with Non-Equijoins:

```
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	G
Vargas	2500	A
Matos	2600	A
Whalen	4400	B
Davies	3100	B
Rajs	3500	B
Lorentz	4200	B
Mourgos	5800	B
Ernst	6000	C

...

20 rows selected.

Outer Joins

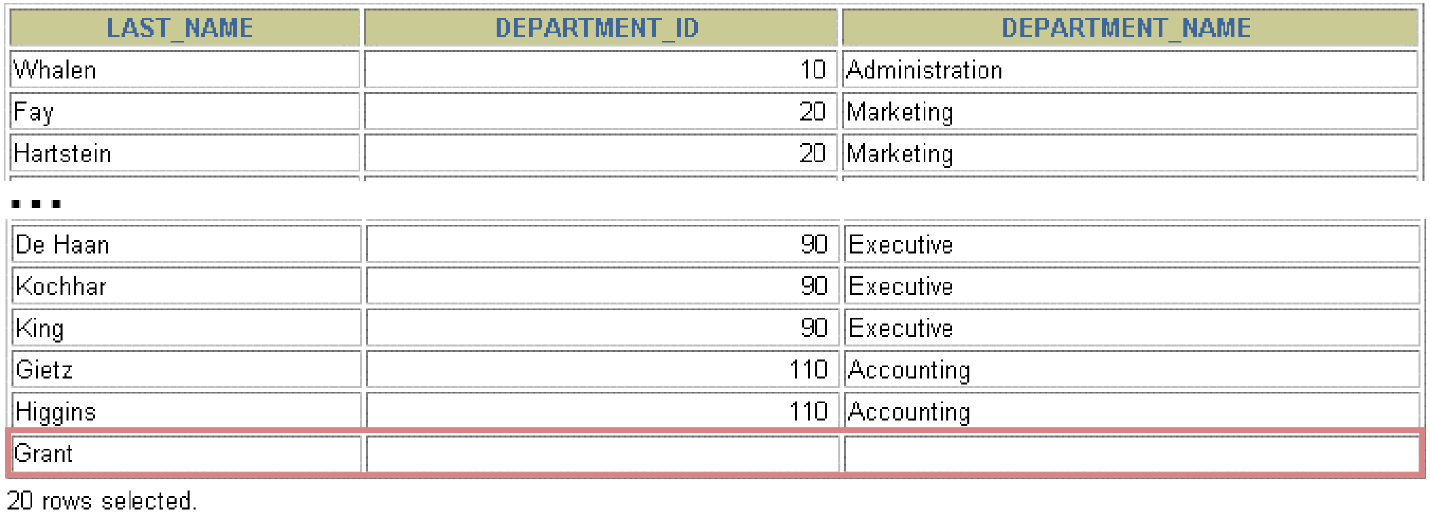
INNER Versus OUTER Joins:

- In SQL:1999, the join of two tables returning only matched rows is called an inner join.
- A join between two tables that returns the results of the inner join as well as the unmatched rows from the left (or right) tables is called a left (or right) outer join.
- A join between two tables that returns the results of an inner join as well as the results of a left and right join is a full outer join.

Outer Joins

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) ;
```



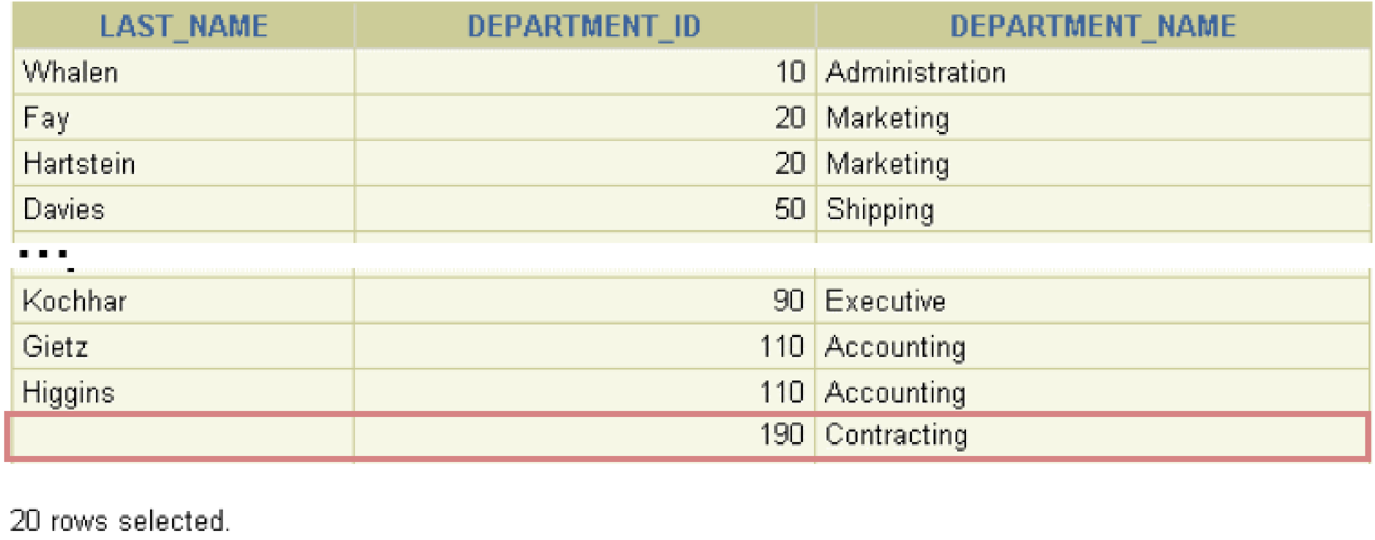
LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Whalen	10	Administration
Fay	20	Marketing
Hartstein	20	Marketing
...		
De Haan	90	Executive
Kochhar	90	Executive
King	90	Executive
Gietz	110	Accounting
Higgins	110	Accounting
Grant		

20 rows selected.

Outer Joins

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
Fay	20	Marketing
Hartstein	20	Marketing
Davies	50	Shipping
...		
Kochhar	90	Executive
Gietz	110	Accounting
Higgins	110	Accounting
	190	Contracting

20 rows selected.

Part 1 Summary

Joins: Presentation

Types of joins

The `ON` Clause

Non-equijoins

Outer Joins