CHAPTER

13

SQL
OPERASI JOIN

Arif Basofi, S.Kom

Information Technology, PENS - ITS

Objectives

Tujuan:

Mengenal perintah SQL dengan operasi **JOIN**:

- Equijoin (Inner Join atau Simple Join)
- Non-Equijoin
- Outer Join (Left Outer Join & Right Outer Join)
- Self Join

S Q L – JOIN

- SQL tidak hanya menyediakan mekanisme query dan operasi modifikasi database saja, tetapi SQL juga menyediakan mekanisme untuk menggabungkan (**join**) relasi-relasi.
- Saat data yang dibutuhkan berasal lebih dari satu table, maka kondisi **<u>join</u>** dibutuhkan.
- Umumnya dalam men-**join** table berdasarkan pada kolom yang bersesuaian **<u>Primary Key</u>** dari table-1dengan **<u>Foreign Key</u>** dari table-2, atau yang disebut dengan **join** atau **equi-join**.
- <u>Kondisi Join meliputi:</u>
  - Equijoin (Inner Join atau Simple Join)
  - Non-Equijoin
  - Outer Join (Left Outer Join & Right Outer Join)
  - Self Join

S Q L – JOIN

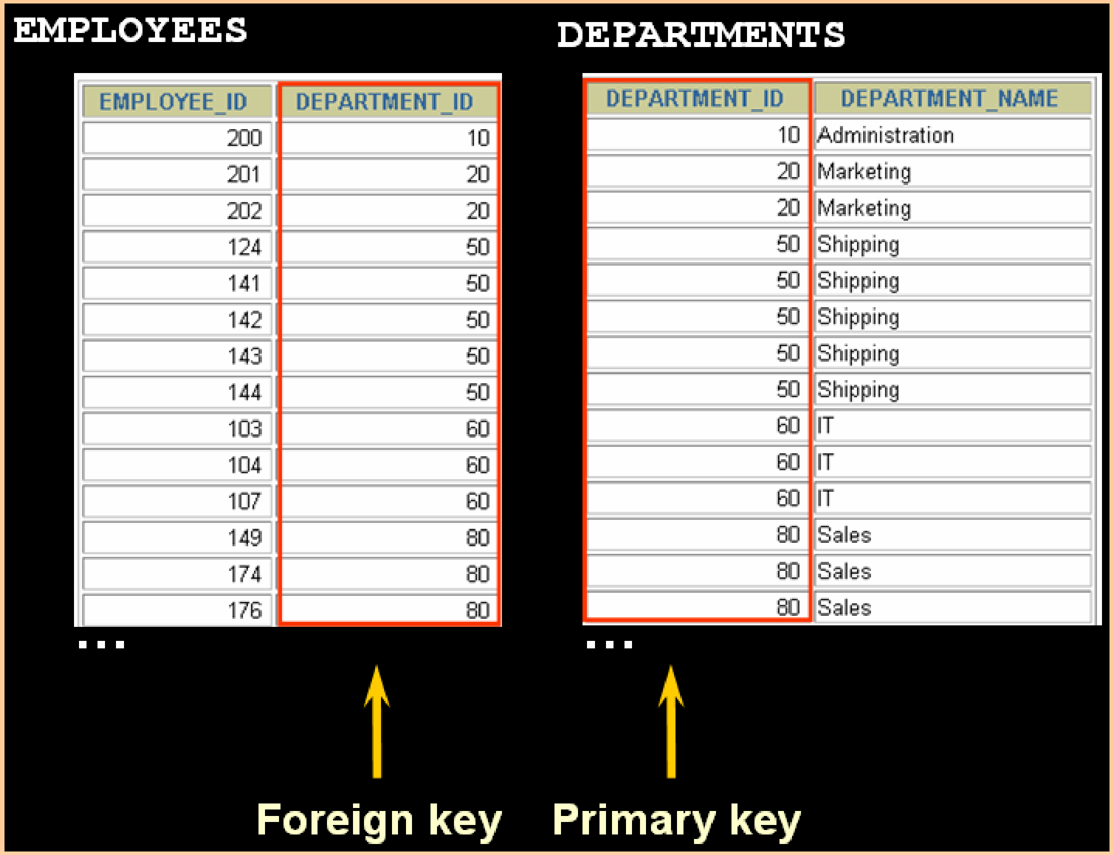
- **Syntax Join SQL JOIN / EQUI-JOIN:**

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

S Q L – EquiJoin (Inner Join atau Simple Join)

- **Join / Equijoin** atau biasa disebut sebagai **Inner Join** atau **Simple Join** adalah bentuk kondisi join dimana nilai relasi yang terjadi antar dua table (binary relation) adalah **sama** (terdapat hubungan antara **Primary Key** dan **Foreign Key**)

Contoh:



EMPLOYEES

EMPLOYEE_ID	DEPARTMENT_ID
200	10
201	20
202	20
124	50
141	50
142	50
143	50
144	50
103	60
104	60
107	60
149	80
174	80
176	80

...

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
20	Marketing
50	Shipping
50	Shipping
50	Shipping
50	Shipping
50	Shipping
60	IT
60	IT
60	IT
80	Sales
80	Sales
80	Sales

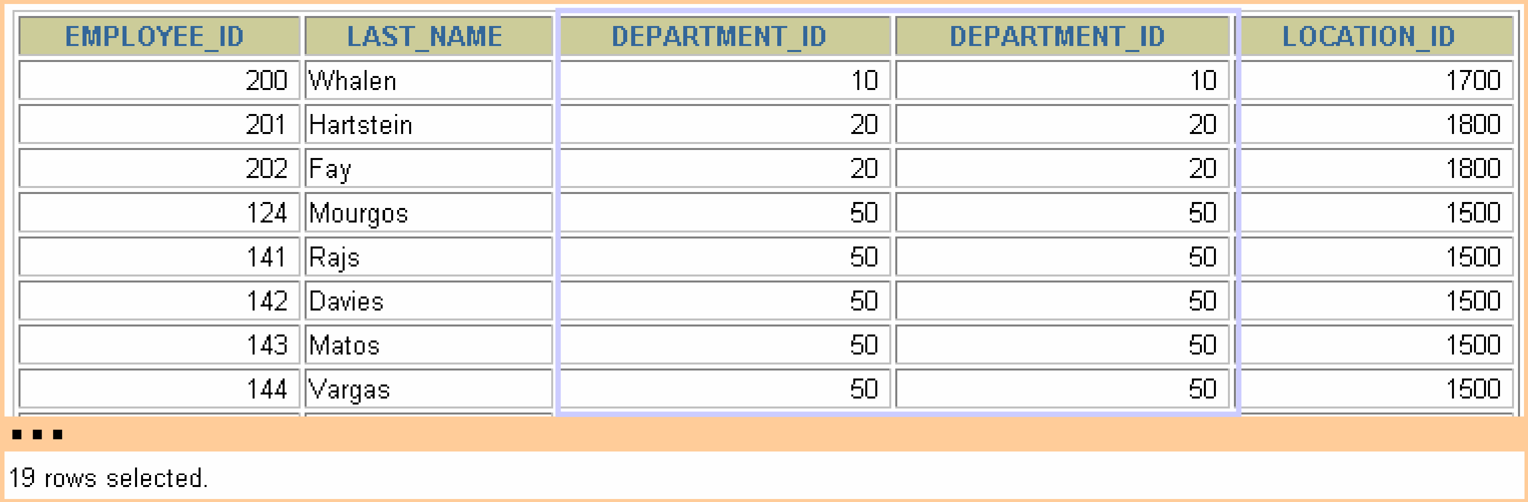
...

Foreign key
Primary key

S Q L – EquiJoin (Inner Join atau Simple Join)

Retrieving Records with Equijoins

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

...

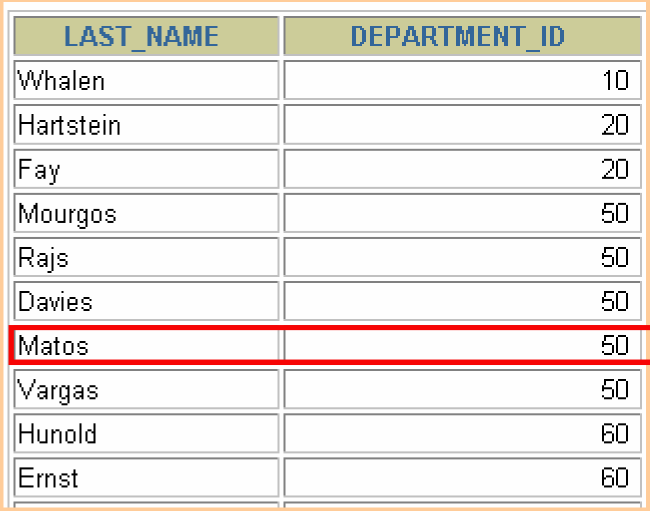
19 rows selected.

S Q L – EquiJoin (Inner Join atau Simple Join)

Kondisi Join dengan Operator AND

```
SELECT last_name, employees.department_id,
           department_name
  FROM   employees, departments
  WHERE  employees.department_id = departments.department_id
  AND    last_name = 'Matos‘;
```

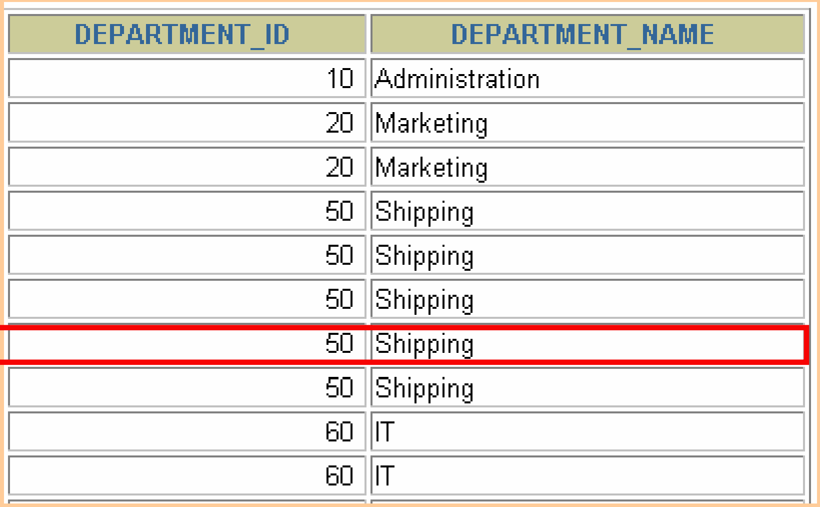
EMPLOYEES



LAST_NAME	DEPARTMENT_ID
Whalen	10
Hartstein	20
Fay	20
Mourgos	50
Rajs	50
Davies	50
Matos	50
Vargas	50
Hunold	60
Ernst	60

...

DEPARTMENTS



DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
20	Marketing
50	Shipping
50	Shipping
50	Shipping
50	Shipping
50	Shipping
60	IT
60	IT

...

S Q L – EquiJoin (Inner Join atau Simple Join)

Kondisi Join Menggunakan Table Alias

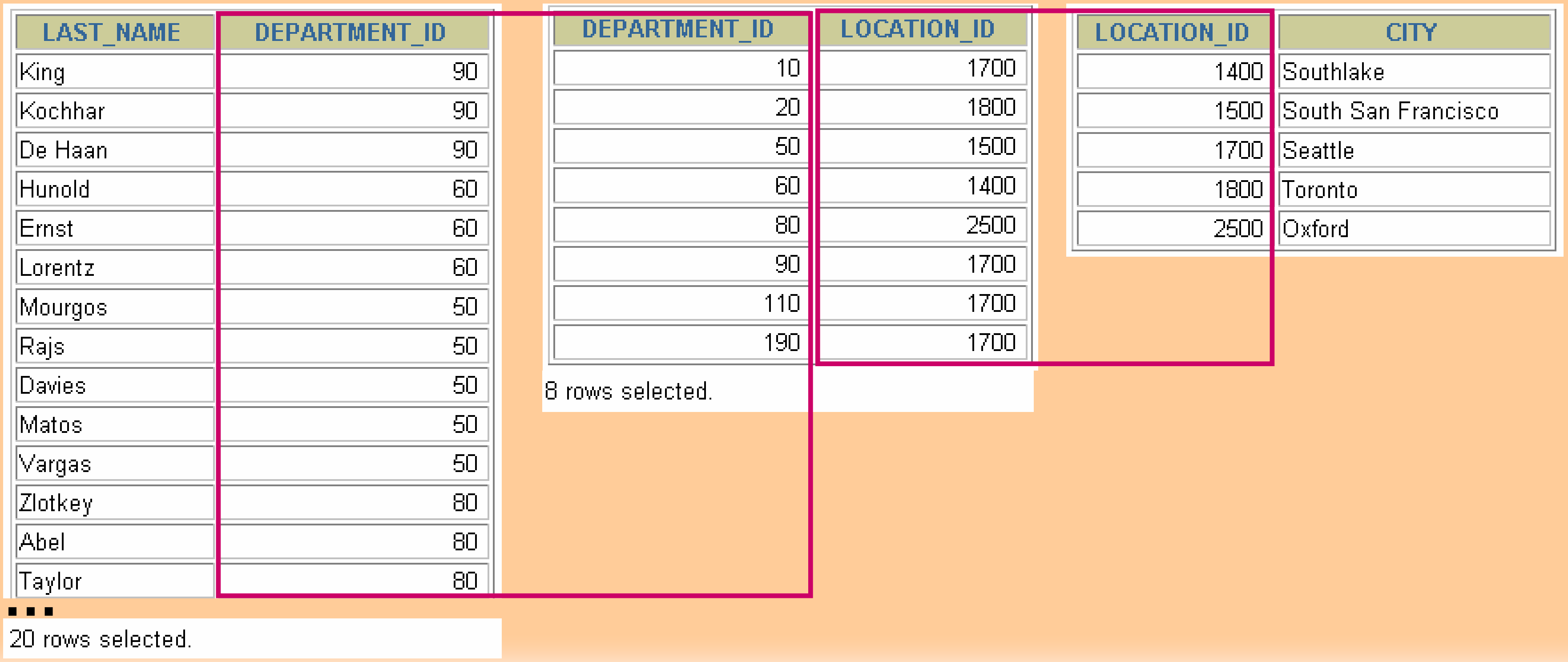
- Menyederhanakan queries dengan menggunakan table alias.
- Meningkatkan performance.

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

S Q L – EquiJoin (Inner Join atau Simple Join)

Kondisi Join Menggunakan Table Alias

```
SELECT e.last_name, d.department_name, l.city
FROM   employees e, departments d, locations l
WHERE  e.department_id = d.department_id
AND    d.location_id = l.location_id;
```



EMPLOYEES

LAST_NAME	DEPARTMENT_ID
King	90
Kochhar	90
De Haan	90
Hunold	60
Ernst	60
Lorentz	60
Mourgos	50
Rajs	50
Davies	50
Matos	50
Vargas	50
Zlotkey	80
Abel	80
Taylor	80

...

20 rows selected.

DEPARTMENTS

DEPARTMENT_ID	LOCATION_ID
10	1700
20	1800
50	1500
60	1400
80	2500
90	1700
110	1700
190	1700

8 rows selected.

LOCATIONS

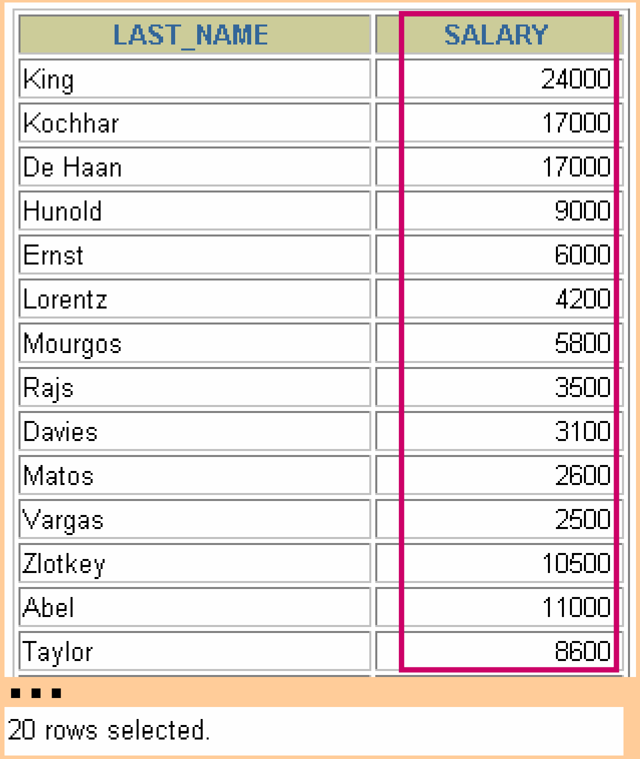
LOCATION_ID	CITY
1400	Southlake
1500	South San Francisco
1700	Seattle
1800	Toronto
2500	Oxford

S Q L – Non - EquiJoin

- **Non - Equijoin** adalah kondisi join yang terkadang tidak mengandung operator sama dengan (=).

Contoh:

EMPLOYEES

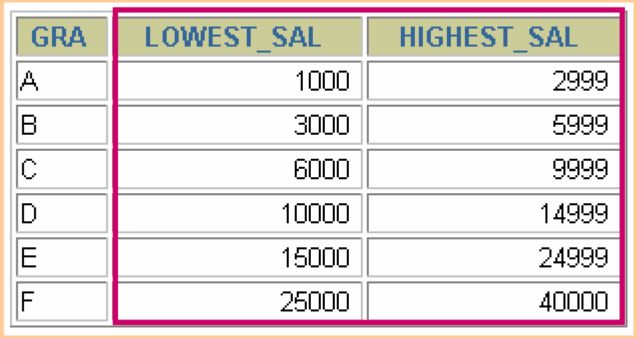


LAST_NAME	SALARY
King	24000
Kochhar	17000
De Haan	17000
Hunold	9000
Ernst	6000
Lorentz	4200
Mourgos	5800
Rajs	3500
Davies	3100
Matos	2600
Vargas	2500
Zlotkey	10500
Abel	11000
Taylor	8600

…

20 rows selected.

JOB_GRADES



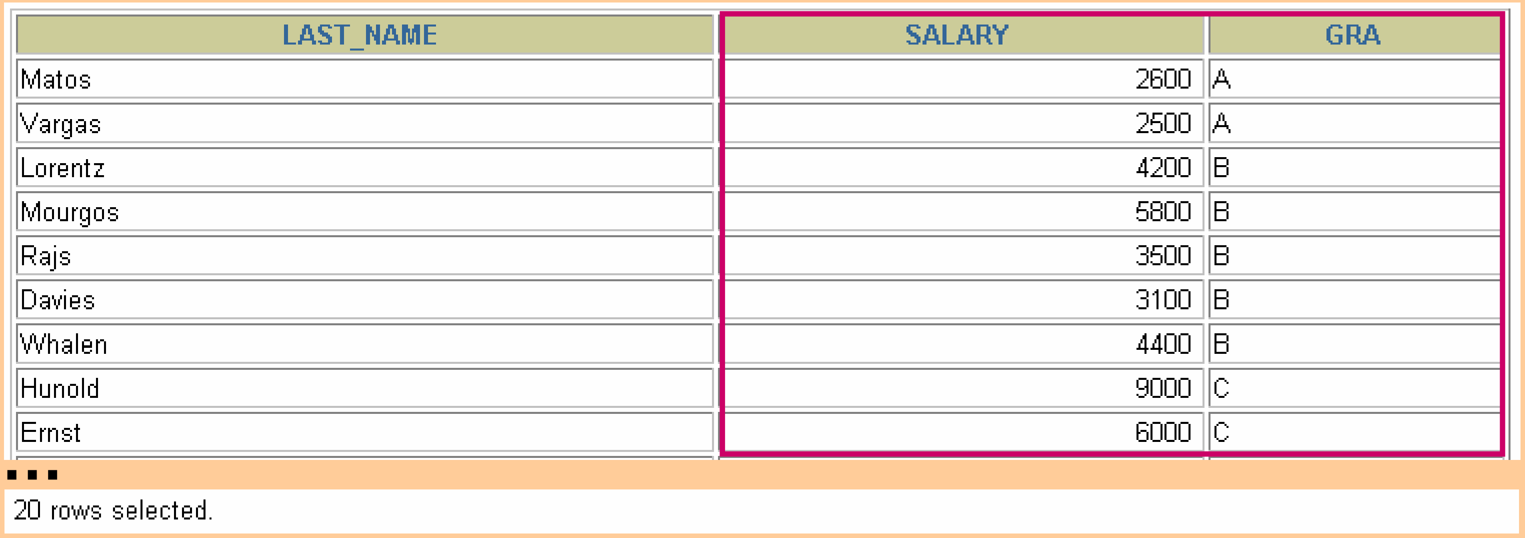
GRA	LOWEST_SAL	HIGHEST_SAL
A	1000	2999
B	3000	5999
C	6000	9999
D	10000	14999
E	15000	24999
F	25000	40000

Salary dalam table EMPLOYEES harus bernilai antara lowest salary dan highest salary pada table JOB_GRADES.

S Q L – Non - EquiJoin

Retrieving Records with Non-Equijoins

```
SELECT e.last_name, e.salary, j.grade_level
FROM   employees e, job_grades j
WHERE  e.salary
       BETWEEN j.lowest_sal AND j.highest_sal;
```



LAST_NAME	SALARY	GRA
Matos	2600	A
Vargas	2500	A
Lorentz	4200	B
Mourgos	5800	B
Rajs	3500	B
Davies	3100	B
Whalen	4400	B
Hunold	9000	C
Ernst	6000	C

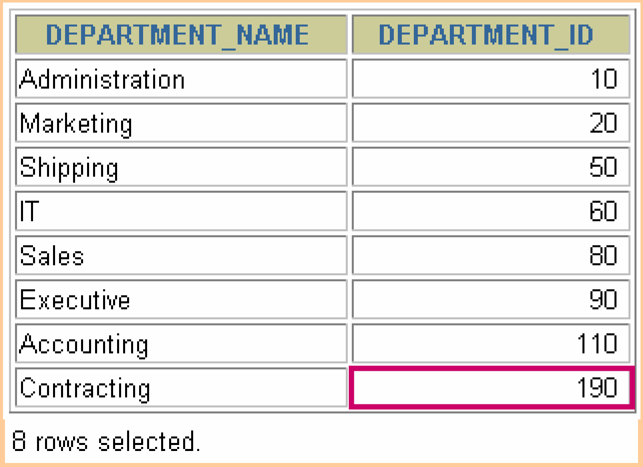
…

20 rows selected.

S Q L – Outer Join

- **<u>Outer Join</u>** adalah bentuk kondisi join untuk mencari nilai join yang memenuhi dari kedua table, plus nilai yang tidak memenuhi dari salah satu sisi table tersebut.
- <u>Contoh</u>: dalam kondisi **<u>equijoin</u>** dari table **EMPLOYEES** dan **DEPARTMENTS**, employee bernama “Grant” tidak muncul, karena memang tidak terdapat id department (dept_id = 190) yang tercatat dari si-“Grant” dalam table **EMPLOYEES**.

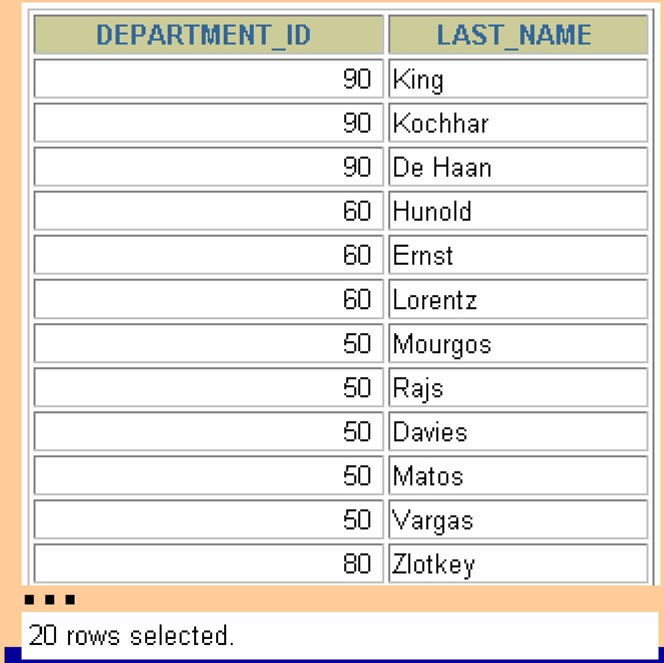
DEPARTMENTS



DEPARTMENT_NAME	DEPARTMENT_ID
Administration	10
Marketing	20
Shipping	50
IT	60
Sales	80
Executive	90
Accounting	110
Contracting	190

8 rows selected.

EMPLOYEES



DEPARTMENT_ID	LAST_NAME
90	King
90	Kochhar
90	De Haan
60	Hunold
60	Ernst
60	Lorentz
50	Mourgos
50	Rajs
50	Davies
50	Matos
50	Vargas
80	Zlotkey

...

20 rows selected.

Tidak ada employees dalam department 190.

S Q L –Outer Join Syntax

- Syntax **Outer Join** ditandai dengan operator **(+).**
- **Outer Join** terdiri atas: **Left Outer Join** dan **Right Outer Join**

Syntax Lengkap Outer Join:

```
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)];
```

S Q L – Left Outer Join Syntax

- Syntax **Outer Join** ditandai dengan operator **(+).**
- **Outer Join** terdiri atas: **Left Outer Join** dan **Right Outer Join**

Left Outer Join

- **Left Outer Join** adalah bentuk join dimana data pada sisi kiri table tidak sempurna / lengkap (yang bertanda (+)) dan akan tertambahkan (dilengkapi) dengan data yang berasal dari sisi kanan table.
- Query akan mendapatkan hasil join semua row yang match (antara table2 dan table1) plus row data dari table 1 yang tidak match.
- Bentuk Syntax Left Outer Join:

```
SELECT table1.column, table2.column
FROM   table1, table2
WHERE  table2.column(+) = table1.column;
```

OR

```
SELECT table1.column, table2.column
FROM   table1
LEFT OUTER JOIN table2
ON   (table1.column = table2.column);
```

S Q L – Right Outer Join Syntax

Right Outer Join

- **Right Outer Join** adalah adalah bentuk join dimana data pada sisi kanan table tidak sempurna / lengkap (yang bertanda (+)) dan akan tertambahkan (dilengkapi) dengan data yang berasal dari sisi kiri table.
- Query akan mendapatkan hasil join semua row yang match (antara table2 dan table1) plus row data dari table 2 yang tidak match.
- Bentuk Syntax Right Outer Join:

```
SELECT table1.column, table2.column
FROM   table1, table2
WHERE  table2.column = table1.column(+);
```

OR

```
SELECT table1.column, table2.column
FROM   table1
Right OUTER JOIN table2
ON    (table1.column = table2.column);
```

S Q L – Contoh Left Outer Join

EMPLOYEES

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	DEPARTMENT_ID
100	Steven	King	90
101	Neena	Kochhar	90
. . . .	. . . .	. . . .	. . .
177	Jack	Livingston	80
178	Kimberely	Grant	
179	Charles	Johnson	80
. . . .	. . . .	. . . .	. . . .
205	Shelley	Higgins	110
206	William	Gietz	110

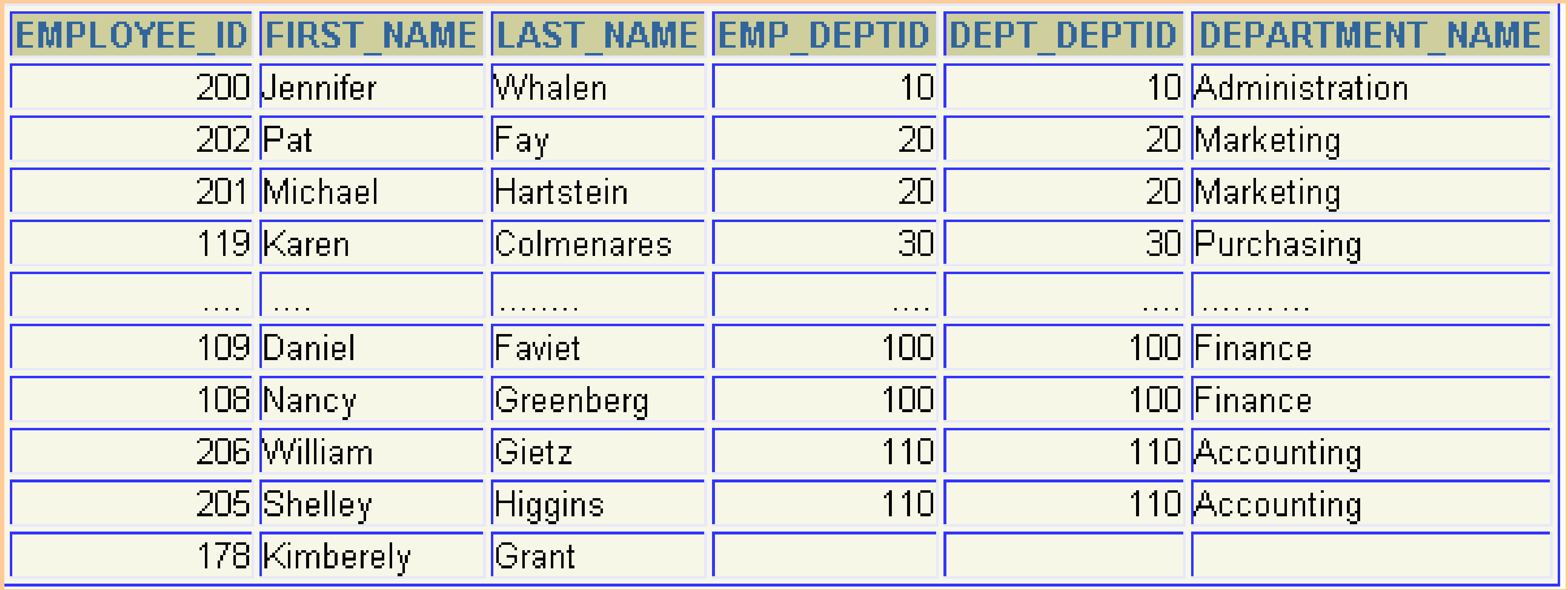
DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
......	.....	....
220	NOC	1700
230	IT Helpdesk	1700
240	Government Sales	1700
250	Retail Sales	1700
260	Recruiting	1700
270	Payroll	1700

```
SELECT e.employee_id,e.first_name,e.last_name,
e.department_id emp_deptid,d.department_id dept_deptid,
d.department_name
FROM employees e ,departments d
WHERE d.department_id (+) = e.department_id
```

S Q L – Contoh Left Outer Join

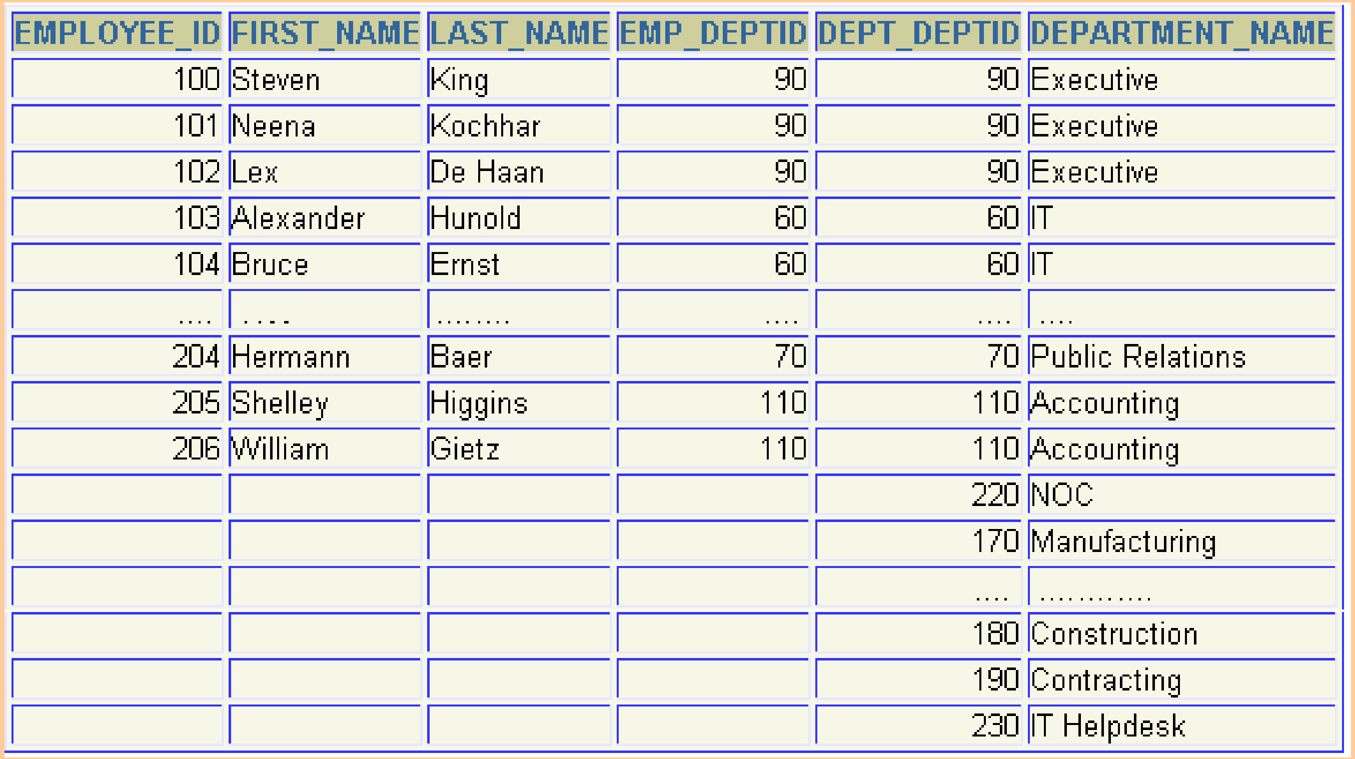
```
SELECT e.employee_id,e.first_name,e.last_name,
e.department_id emp_deptid,d.department_id dept_deptid,
d.department_name
FROM employees e ,departments d
WHERE d.department_id (+) = e.department_id
```



EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMP_DEPTID	DEPT_DEPTID	DEPARTMENT_NAME
200	Jennifer	Whalen	10	10	Administration
202	Pat	Fay	20	20	Marketing
201	Michael	Hartstein	20	20	Marketing
119	Karen	Colmenares	30	30	Purchasing
….	….	……..	….	….	……… …
109	Daniel	Faviet	100	100	Finance
108	Nancy	Greenberg	100	100	Finance
206	William	Gietz	110	110	Accounting
205	Shelley	Higgins	110	110	Accounting
178	Kimberely	Grant			

S Q L – Contoh Right Outer Join

```
SELECT e.employee_id,e.first_name,e.last_name,
e.department_id emp_deptid,d.department_id dept_deptid,
d.department_name
FROM employees e ,departments d
WHERE d.department_id=e.department_id (+)
```

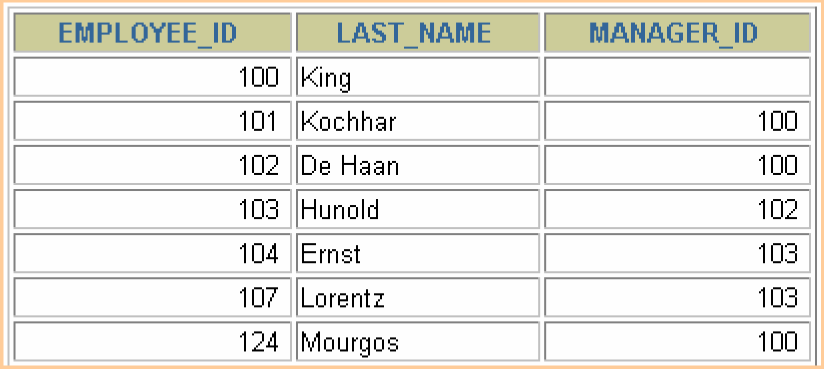


EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMP_DEPTID	DEPT_DEPTID	DEPARTMENT_NAME
100	Steven	King	90	90	Executive
101	Neena	Kochhar	90	90	Executive
102	Lex	De Haan	90	90	Executive
103	Alexander	Hunold	60	60	IT
104	Bruce	Ernst	60	60	IT
....	....	....	....	....	....
204	Hermann	Baer	70	70	Public Relations
205	Shelley	Higgins	110	110	Accounting
206	William	Gietz	110	110	Accounting
				220	NOC
				170	Manufacturing
				....	....
				180	Construction
				190	Contracting
				230	IT Helpdesk

S Q L – Self Join

- **Self Join** adalah bentuk kondisi join yang terjadi pada table diri sendiri (recursive).
- Misal. Ingin mencari nama manager dari tiap employee, tentunya akan mencari pada table yang sama yaitu **EMPLOYEES**.

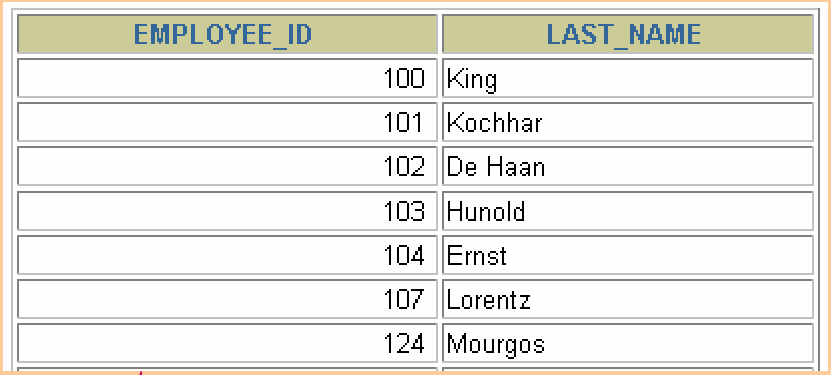
EMPLOYEES (WORKER)



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

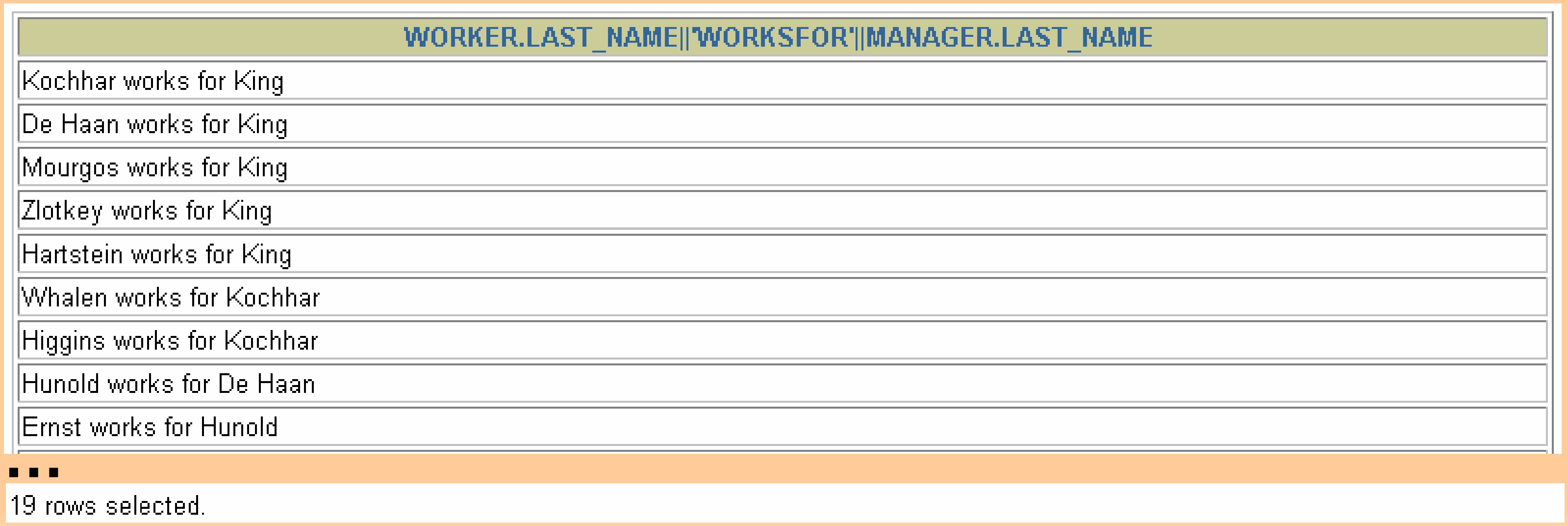
...

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

S Q L – Self Join

- **Self Join** adalah bentuk kondisi join yang terjadi pada table diri sendiri (recursive).
- Misal. Ingin mencari nama manager dari tiap employee, tentunya akan mencari pada table yang sama yaitu **EMPLOYEES**.

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



```
WORKER.LAST_NAME||'WORKSFOR'||MANAGER.LAST_NAME
Kochhar works for King
De Haan works for King
Mourgos works for King
Zlotkey works for King
Hartstein works for King
Whalen works for Kochhar
Higgins works for Kochhar
Hunold works for De Haan
Ernst works for Hunold
...
19 rows selected.
```

S Q L – Excercise

Latihan:

1. Buat SQL Query untuk menampilkan last name, department number, dan department name untuk semua pegawai.
2. Tampilkan semua job (job id) pegawai secara unik yang berada pada department 80 termasuk nama lokasinya.
3. Buat query yang menampilkan last name, nama department, location id dan kota dari semua pegawai yang memiliki komisi.
4. Tampilkan last name pegawai dan nama department untuk semua pegawai yang memiliki huruf ‘a’ pada last name.
5. Buat query yang menampilkan last name, department number, department name untuk semua pegawai yang bekerja di kota Toronto.

Latihan

- Buat sistem database sederhana menggunakan Microsoft Access.
- Rancang tabel-tabel datase beserta relasinya.
- Masukkan data pada tiap tabel.
- Lakukan operasi SQL Query pada tabel.