BAB 5 : Menampilkan Data dari Banyak Tabel

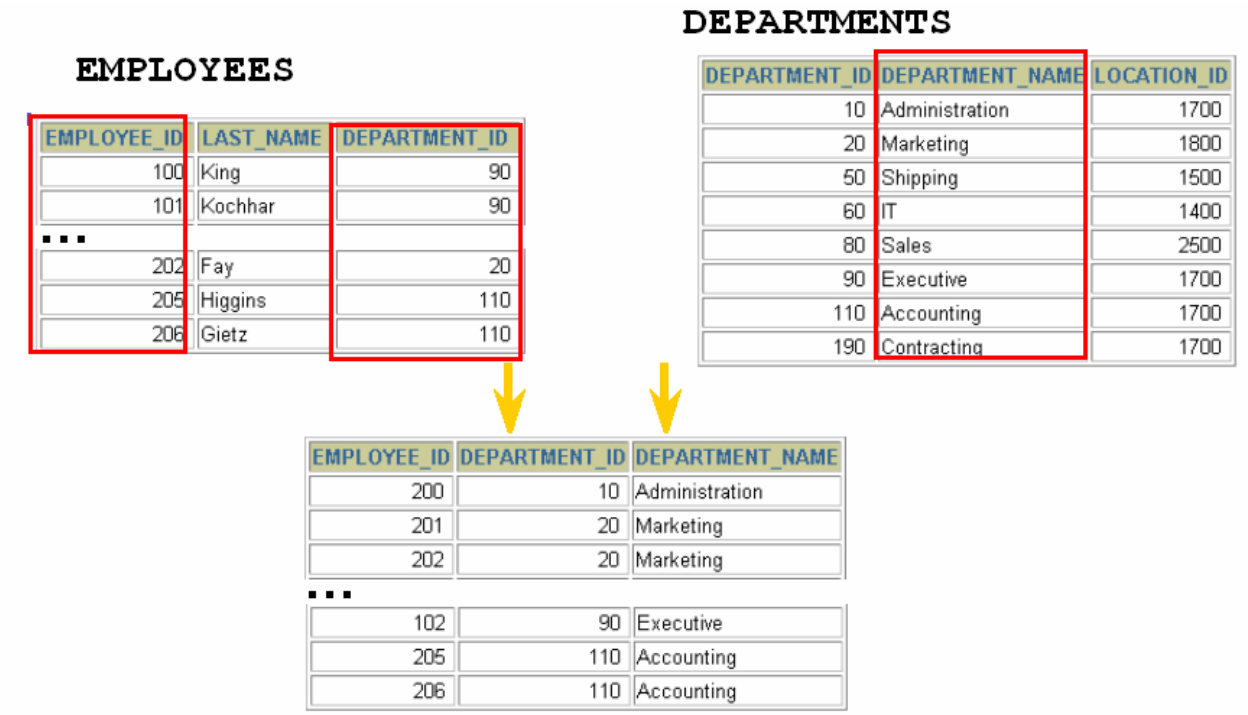
5.1. Sasaran

- Dapat menulis statement SELECT yang mengakses data ke lebih dari satu table dengan menggunakan operator join.
- Menampilkan data yang tidak memenuhi kondisi join dengan menggunakan operator outer join
- Melakukan join terhadap table itu sendiri (self join)

5.2. Mendapatkan Data dari Banyak Tabel

Seringkali kita perlu menggunakan data yang berasal dari banyak table, tidak hanya berasal dari satu table saja, semisal :

- nomer pegawai (employee_id) hanya ada di table pegawai (EMPLOYEES)
- nomer department (department_id) ada di table pegawai (EMPLOYEES) dan table department (DEPARTMENTS).
- Nama departemen (Department_name) hanya ada di table department (DEPARTMENTS)



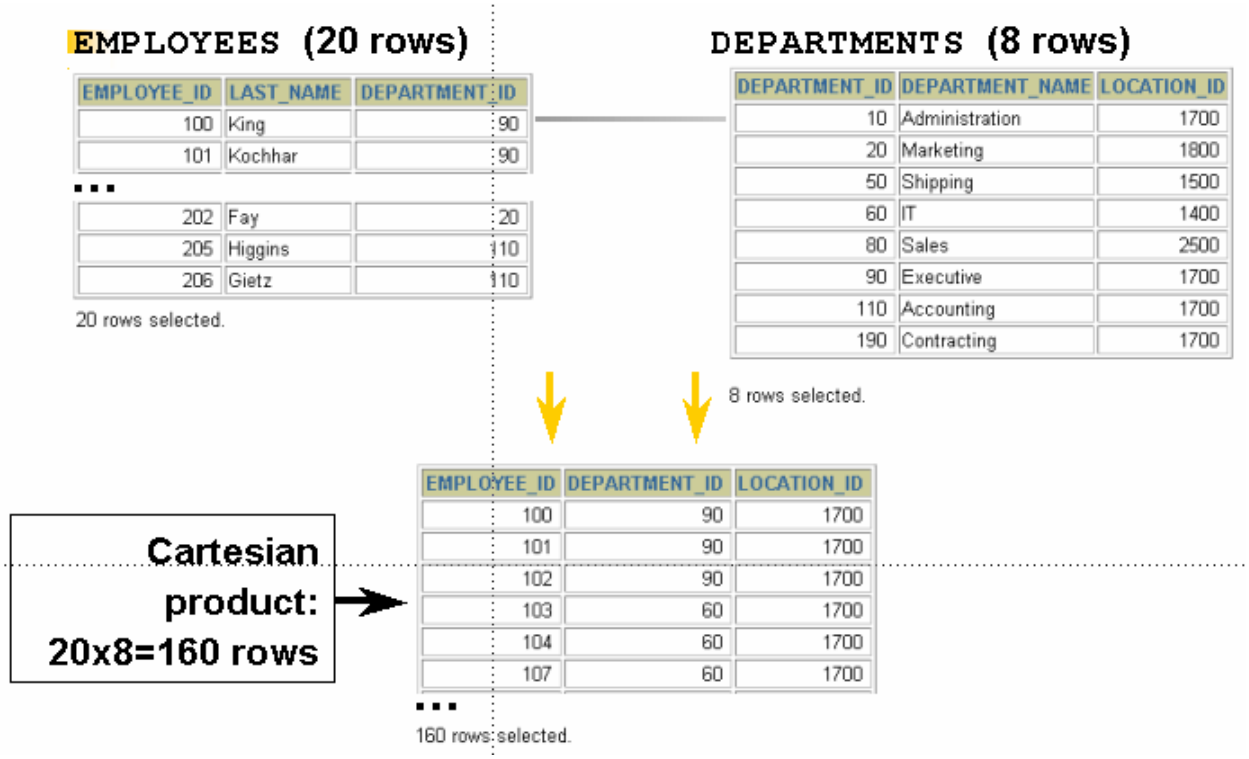
5.3. Cartesian Products

Cartesian product dibentuk pada saat :

- Kondisi join diabaikan
- Kondisi join tidak valid
- Semua baris dalam table pertama dijoinkan ke semua baris dalam table kedua

5.4. Mengenerate Cartesian Product

Karena Cartesian Product meliputi semua kombinasi data dari dua tabel, maka tabel hasilnya berisi sejumlah perkalian antara jumlah record dari tabel pertama dikalikan dengan jumlah record tabel kedua.



5.5. Tipe-tipe Join

Berikut tipe-tipe join (standart SQL 92) yang dipunyai oleh versi Oracle8i keatas :

- Equijoin
- Non-equijoin
- Outer join
- Self join

Berikut tipe-tipe join tambahan (standart SQL 99) yang ada mulai Oracle9i :

- Cross join
- Natural join
- Klausa Using
- Full or two sided Outer Join
- Kondisi Join untuk Outer Join

5.6. Dua tipe Utama Join

Ada 2 (dua) tipe utama join, yaitu equi-join dan non-equijoin.
Metode join lainnya pada standart SQL92 yaitu meliputi : outer join dan self join.

5.7. EquiJoin

Misal table EMPLOYEES memiliki primary key *employee_id*, dan memiliki foreign key *department_id* dimana *departement_id* ini merupakan primary key dari table yang lain yaitu table DEPARTMENTS. Relasi antara EMPLOYEES dengan DEPARTEMENTS disebut equi-join.
Relasi antara dua tabel ditulis dalam klausa WHERE. Dari kedua tabel pada contoh kasus tersebut, maka jika kita ingin menampilkan nomer pegawai, nomer departemen dan nama departemennya, maka perintah SQL yang dibuat :

```
SELECT employee_id, department_id, department_name
FROM EMPLOYEES, DEPARTMENTS
WHERE EMPLOYEES.Department_id = DEPARTMENTS.Department_id;
```

5.8. Mendapatkan Record dengan EquiJoin

Berikut contoh mendapatkan record data dengan menggunakan EquiJoin :

```
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.

5.9. Menambahkan Kondisi Pencarian dengan Operator AND

Operator logika (AND, OR, NOT) bisa digunakan pada kondisi pencarian yang ada pada klausa WHERE.

Jika pada statement SQL sebelumnya (5.8) ditambahkan kondisi pencarian untuk nama pegawai = 'Matos' saja yang akan ditampilkan maka perintahnya SQL nya :

```
SELECT employees.employee_id, employees.last_name, employees_department_id,
       departments.department_id, departements.location_id
FROM employees, departments
WHERE employees.department_id = departments.department_id AND
employees.last_name='Matos';
```

5.10. Penggunaan Tabel Alias

Query dapat disederhanakan dengan penggunaan table alias.

Contoh query berikut :

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

Dengan menggunakan table alias akan diubah seperti berikut :

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

Penggunaan tabel alias juga digunakan untuk nama kolom yang ambigu artinya nama kolom yang sama dimiliki oleh lebih dari satu tabel.

5.11. Men-Join-kan lebih dari Dua Tabel

Berikut contoh ilustrasi join lebih dari dua tabel :

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

Relasi dari ketiga tabel dinyatakan dalam klausa WHERE sebagai berikut :

```
WHERE employees.department_id = departments.department_id AND
      departments.location_id = locations.location_id;
```

5.12. Non-EquiJoin

Relasi antara dua table disebut non-equijoin jika kolom pada table pertama berkorespondensi langsung dengan kolom pada table kedua.

Berikut ini contoh yang mengilustrasikan non-equijoin :

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 pada tabel EMPLOYEES harus berada diantara nilai salary terendah dengan nilai salary tertinggi yang ada pada tabel JOB_GRADES.

5.13. Mendapatkan Record dengan Non-EquiJoin

```
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.

5.14. Outer Join

Jika terdapat baris yang tidak memenuhi kondisi join, dan akan ditampilkan pada hasil query, maka digunakan *outer join.*
Misal pada hasil query berikut, nama departemen 'CONTRACTING' tidak ditampilkan karena tidak memenuhi kondisi join, artinya pada tabel employee tidak ada pegawai yang bekerja pada departemen CONTRACTING.

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 pegawai yang bekerja di department 190.

5.15. Sintak dari Outer Join

Operator Outer Join adalah tanda plus (+). Sintak dari Outer Join :

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

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

5.16. Penggunaan Outer Join

Jika akan ditampilkan kolom pada tabel departemen (DEPARTMENTS) yang tidak bersesuaian dengan semua kolom yang ada pada table pegawai (EMPLOYEES), (**dalam kondisi :** tidak ada pegawai yang berkerja di departemen 'CONTRACTING' dengan nomer 190, sehingga nomer 190 tidak muncul di tabel employees), digunakan query dengan outer join berikut :

```
SELECT e.last_name, e.department_id, d.department_name
FROM   employees e, departments d
WHERE  e.department_id(+) = d.department_id ;
```

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Whalen	10	Administration
Hartstein	20	Marketing
Fay	20	Marketing
Mourgos	50	Shipping
Rajs	50	Shipping
Davies	50	Shipping
Matos	50	Shipping
...		
Gietz	110	Accounting
		Contracting

20 rows selected.

5.17. Self Join

Seringkali sebuah table perlu dijoin-kan dengan table itu sendiri. Misal pada saat mencari manager dari seorang pegawai maka table pegawai di-joinkan dengan table pegawai untuk

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 pada tabel WORKER sama dengan EMPLOYEE_ID pada tabel MANAGER.

mendapatkan nomer pegawai manager dan namanya.

5.18. Men-Join-kan Tabel ke tabel itu sendiri

```
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.

5.19. Latihan 5, Bagian 1

1. Buat query untuk menampilkan nama pegawai, nomer department dan nama department dari semua pegawai

```
ENAME           DEPTNO DNAME
---------- ---------- --------------
SMITH               20 RESEARCH
ALLEN               30 SALES
WARD                30 SALES
JONES               20 RESEARCH
MARTIN              30 SALES
BLAKE               30 SALES
CLARK               10 ACCOUNTING
```

…….

2. Buat daftar yang unik dari semua pekerjaan pada department 30, tampilkan pula lokasi dari department 30 pada output.

```
JOB       LOC
--------- -------------
CLERK     CHICAGO
MANAGER   CHICAGO
SALESMAN  CHICAGO
```

3. Tampilkan nama pegawai, nama department dan lokasi dari semua pegawai yang memiliki komisi (komisi tidak sama dengan NULL)

```
ENAME      DNAME          LOC
---------- -------------- -------------
ALLEN      SALES          CHICAGO
WARD       SALES          CHICAGO
MARTIN     SALES          CHICAGO
```

4. Tampilkan nama pegawai dan nama department untuk semua pegawai yang memiliki huruf ‘A’ pada namanya.

```
ENAME      DNAME
---------- -------------
ALLEN      SALES
WARD       SALES
MARTIN     SALES
BLAKE      SALES
CLARK      ACCOUNTING
ADAMS      RESEARCH
JAMES      SALES

7 rows selected.
```

5. Buat query untuk menampilkan nama pegawai, pekerjaan, nomer department, dan nama department untuk semua pegawai yang bekerja di kota ‘DALLAS’

```
ENAME      JOB            DEPTNO DNAME
---------- --------- ---------- -----------
SMITH      CLERK              20 RESEARCH
JONES      MANAGER            20 RESEARCH
SCOTT      ANALYST            20 RESEARCH
ADAMS      CLERK              20 RESEARCH
FORD       ANALYST            20 RESEARCH
```

6. Buat query untuk menampilkan nama pegawai dan nomer pegawai, nama manager dan nomer pegawai dari manager.

```
  EMPNO PEGAWAI         EMPNO MANAGER
------- ---------- ---------- ----------
   7369 SMITH            7902 FORD
   7499 ALLEN            7698 BLAKE
   7521 WARD             7698 BLAKE
   7566 JONES            7839 KING
   7654 MARTIN           7698 BLAKE
   7698 BLAKE            7839 KING
..........
      7902 FORD             7566 JONES
      7934 MILLER           7782 CLARK

13 rows selected.
```

7. Modifikasi query pada nomer 6, buat outer join untuk menampilkan pula data pegawai yang tidak mempunyai manager.

```
     EMPNO PEGAWAI         EMPNO MANAGER
---------- ---------- ---------- ---------
      7369 SMITH            7902 FORD
      7499 ALLEN            7698 BLAKE
      7521 WARD             7698 BLAKE
      7566 JONES            7839 KING
      7654 MARTIN           7698 BLAKE
      7698 BLAKE            7839 KING
      7782 CLARK            7839 KING
      7788 SCOTT            7566 JONES
      7839 KING
      7844 TURNER           7698 BLAKE
.........
      7902 FORD             7566 JONES
      7934 MILLER           7782 CLARK

14 rows selected.
```

8. Buat query yang menampilkan nama pegawai, nomer department, dan semua employee yang bekerja pada department yang sama dengan employee. Samakan judul kolom seperti yang ada pada hasil berikut :

```
DEPARTMENT PEGAWAI    KOLEGA
---------- ---------- ----------
        10 KING       CLARK
        10 MILLER     CLARK
        10 CLARK      KING
        10 MILLER     KING
        10 CLARK      MILLER
        10 KING       MILLER
.........
        30 TURNER     WARD

56 rows selected.
```

9. Tampilkan struktur dari table SALGRADE. Buat query yang menampilkan nama pegawai , pekerjaan, nama department, gaji dan grade untuk semua pegawai

ENAME	JOB	DNAME	SAL	GRADE
SMITH	CLERK	RESEARCH	800	1
ADAMS	CLERK	RESEARCH	1100	1
JAMES	CLERK	SALES	950	1
WARD	SALESMAN	SALES	1250	2
MARTIN	SALESMAN	SALES	1250	2
MILLER	CLERK	ACCOUNTING	1300	2
ALLEN	SALESMAN	SALES	1600	3

……..

10. Buat query untuk menampilkan nama dan tanggal mulai bekerja dari pegawai yang tanggal bekerjanya setelah pegawai bernama 'BLAKE'

ENAME	HIREDATE
MARTIN	28-SEP-81
CLARK	09-JUN-81
SCOTT	19-APR-87
KING	17-NOV-81
TURNER	08-SEP-81
ADAMS	23-MAY-87
JAMES	03-DEC-81
FORD	03-DEC-81
MILLER	23-JAN-82

9 rows selected.

11. Tampilkan semua nama pegawai dan tanggal kerjanya serta nama manager dan tanggal kerjanya dimana tanggal mulai kerja pegawai lebih dulu daripada tanggal mulai kerja managernya.

PEGAWAI	HIREDATE	MANAGER	HIREDATE
SMITH	17-DEC-80	FORD	03-DEC-81
ALLEN	20-FEB-81	BLAKE	01-MAY-81
WARD	22-FEB-81	BLAKE	01-MAY-81
JONES	02-APR-81	KING	17-NOV-81
BLAKE	01-MAY-81	KING	17-NOV-81
CLARK	09-JUN-81	KING	17-NOV-81

6 rows selected.

5.19. Men-Join-kan Tabel dengan Sintak SQL 99

Berikut ini sintak atau bentuk umum penulisan join table dengan SQL 99 :

```
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)];
```

5.20. Pembuatan Cross Join

Klausa CROSS JOIN menghasilkan cross-product dari dua table, ini berarti sama dengan Cartesian product yang dihasilkan oleh dua table.

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

LAST_NAME	DEPARTMENT_NAME
King	Administration
Kochhar	Administration
De Haan	Administration
Hunold	Administration

...

160 rows selected.

SQL Statement diatas sama dengan perintah SQL92 berikut ini :

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

5.21. Pembuatan Natural Join

Klausa NATURAL JOIN dibuat berdasarkan semua kolom pada dua table yang memiliki nama yang sama. Baris terpilih adalah yang memiliki nilai yang sama untuk setiap kolom yang bersesuaian dari dua table. Jika kolom memiliki nama yang sama tapi tipe data berbeda, maka akan terjadi error.

```
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
110	Accounting	1700	Seattle
190	Contracting	1700	Seattle
20	Marketing	1800	Toronto
80	Sales	2500	Oxford

8 rows selected.

Perintah JOIN diatas sama dengan :

```
SELECT department_id, department_name,
      location_id, city
FROM   departments.location_id = locations.location_id ;
```

5.22. Pembuatan Join dengan Klausa USING

Jika beberapa kolom memiliki nama yang sama tapi tipe datanya tidak sesuai maka NATURAL JOIN dapat diubah dengan menggunakan klausa USING untuk menentukan kolom mana yang harus digunakan. Klausa USING digunakan hanya untuk mencocokkan satu kolom saja pada saat lebih dari

satu kolom yang sesuai. Tidak diperbolehkan untuk menggunakan nama table atau alias dalam kolom referensi.

```
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
124	Mourgos	1500
141	Rajs	1500
142	Davies	1500
143	Matos	1500
144	Vargas	1500
103	Hunold	1400

...

19 rows selected.

Perintah SQL diatas sama hasilnya dengan :

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

5.23. Pembuatan Join dengan Klausa ON

Kondisi JOIN untuk natural join dibuat berdasarkan equijoin dari semua kolom yang memiliki nama yang sama. Untuk menentukan kondisi atau menentukan kolom mana yang perlu dijoinkan, maka digunakan klausa ON. Dengan demikian maka klausa ON juga dapat dipakai untuk kolom yang memiliki nama yang tidak sama, dan kondisi JOIN terpisah dari kondisi pencarian. Dengan menggunakan klausa ON, membuat SQL Statement menjadi lebih mudah dipahami.

```
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.

Perintah SQL diatas, sama hasilnya dengan :

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

5.24. Pembuatan Three-Way Join dengan Klausa ON

Berikut ini join tiga table dengan menggunakan klausa 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
103	Southlake	IT
104	Southlake	IT
107	Southlake	IT
124	South San Francisco	Shipping
141	South San Francisco	Shipping
142	South San Francisco	Shipping
143	South San Francisco	Shipping
144	South San Francisco	Shipping

...

19 rows selected.

5.25. INNER vs. OUTER JOIN

Pada SQL 99, join dari dua table hanya mengembalikan nilai baris yang sesuai (inner join), sementara kadang perlu ditampilkan data dari kondisi join yang tidak match. Misal jika akan ditampilkan data dari table employees dan departments, sedangkan kondisi joinnya adalah department_id dari masing-masing table. Jika terdapat data department yang tidak memiliki karyawan, dalam arti terdapat nomer department (department_id) pada table departments yang tidak muncul sama sekali pada department_id yang ada pada table employees, maka untuk memunculkannya kita perlu outer join. Outer Join ada yang right atau left tergantung *unmatched rows* dari table mana kita mengambilnya.

5.26. LEFT OUTER JOIN

Berikut contoh 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.

5.27. RIGHT OUTER JOIN

Berikut ini contoh 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
King	90	Executive
Kochhar	90	Executive
...		
Whalen	10	Administration
Hartstein	20	Marketing
Fay	20	Marketing
Higgins	110	Accounting
Gietz	110	Accounting
		Contracting

20 rows selected.

5.28. FULL OUTER JOIN

Berikut ini contoh 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
...		
De Haan	90	Executive
Kochhar	90	Executive
King	90	Executive
Gietz	110	Accounting
Higgins	110	Accounting
Grant		
		Contracting

21 rows selected.

5.29. Kondisi Tambahan

Kondisi tambahan yang diberikan pada JOIN, dapat dengan menggunakan operator logika AND.

```
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

5.34. Latihan 4, Bagian 2

Petunjuk : gunakan user HR.

1. Tulis query untuk menampilkan last name, job, nomer department dan nama department untuk semua pegawai yang bekerja di Toronto
2. Tamplkan last name dan nomer employee serta last name dan nomer employee dari manager. Beri judul Employee, Emp#, Manager dan Mgr#.
3. Modifikasi query pada soal nomer 2 dengan menampilkan pegawai bernama King yang tidak memiliki manager.
4. Buat query untuk menampilkan data pegawai yaitu last name, nomer department, dan semua pegawai yang bekerja di department yang sama dengan pegawai tersebut.
5. Buat query yang menampilkan nama dan tanggal bekerja dari pegawai yang tanggal kerjanya setelah pegawai yang bernama Davies.
6. Tampilkan nama dan tanggal kerja dari semua pegawai yang tanggal mulai bekerjanya sebelum manager dari pegawai tersebut.