7. MENAMPILKAN DATA DARI BANYAK TABEL

Objektif

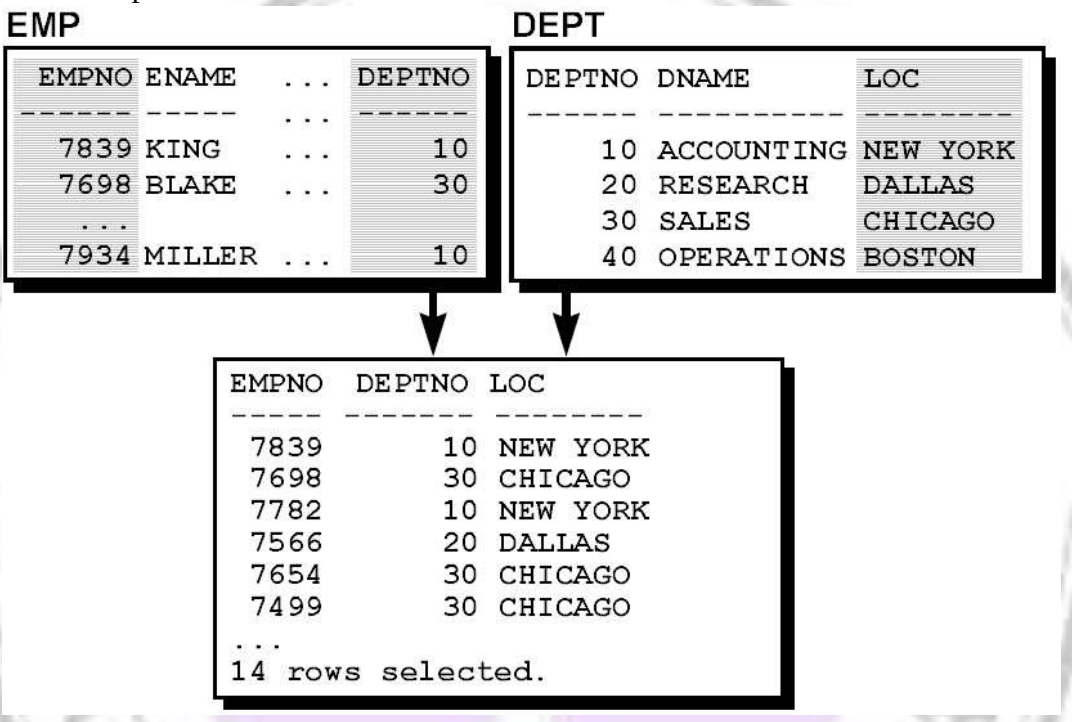
Setelah mengikuti materi ini, diharapkan untuk dapat :

1. Menulis statement SELECT untuk mengakses data lebih dari satu tabel dengan menggunakan *Equijoin* dan *Non-Equijoin.*
2. Menampilkan data yang secara umum tidak diperoleh pada kondisi *join* dengan menggunakan *outer join.*
3. Melakukan join terhadap table itu sendiri (self join).

5.1. Mendapatkan Data Dari Banyak Tabel

Terkadang diperlukan data yang berasal dari banyak table, tidak hanya berasal dari satu table saja, contoh:

- EMPNO terdapat di dalam tabel EMP
- DEPTNO terdapat di dalam table EMP dan table DEPT.
- LOC terdapat di dalam table DEPT.



5.2. Definisi Join

Penggunaan kondisi join dilakukan jika membutuhkan data dari salah satu tabel dalam database. Baris dalam suatu tabel dapat join ke baris tabel lain menurut nilai yang ada pada kolom yang berhubungan (sesuai), umumnya adalah kolom primary key dan foreign key.

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

Sintaks :

Table1.column	menunjukkan tabel dan kolom dari data yang diinginkan
Tabel1.column1	adalah kondisi dimana tabel di join (atau dihubungkan) bersama-sama tabel2.column2

5.3. Cartesian Product

Cartesian product dibentuk pada saat:

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

Untuk menghindari Cartesian product, gunakan selalu klausa WHERE dalam kondisi join.

5.4. Membentuk Cartesian Product

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.

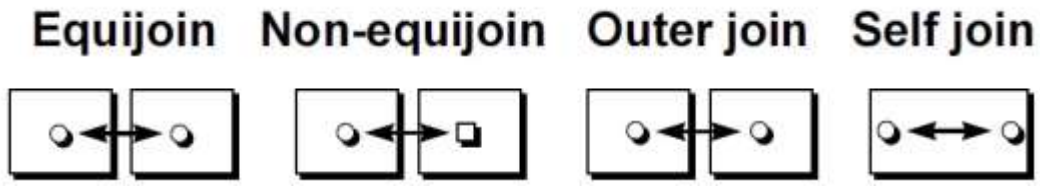
Contoh menampilkan ENAME dari table EMP dan DNAME dari table DEPT dengan membentuk Cartesian Product:

```
SQL> SELECT    emp.ename, dept.dname,
  2 FROM       emp, dept;
```

Lihat Video

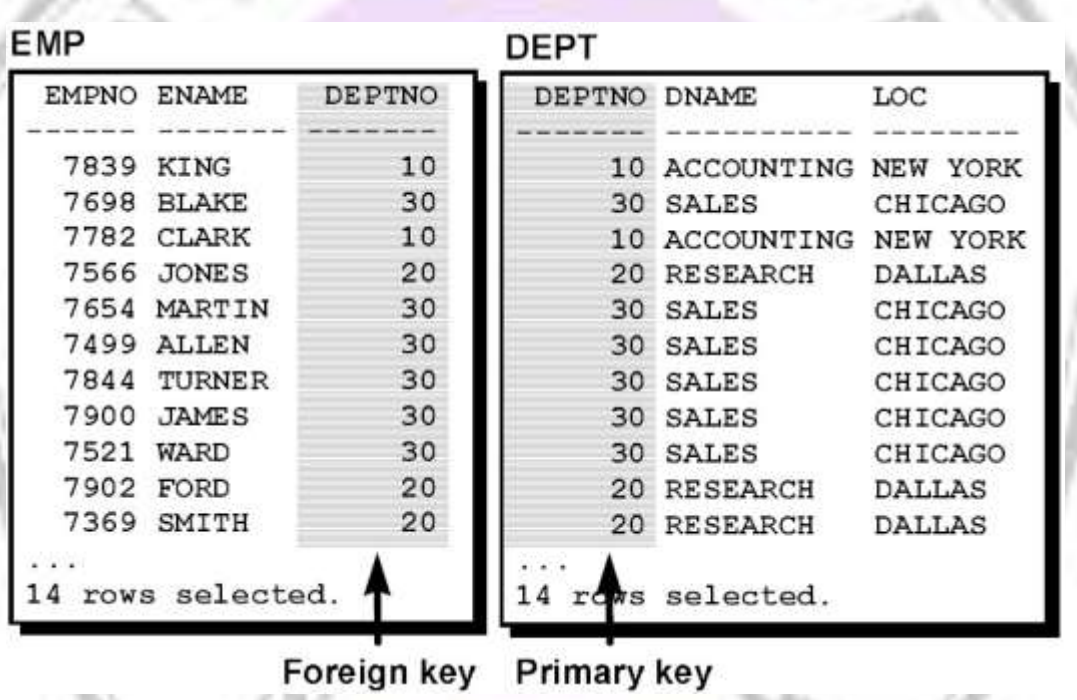
5.5. Tipe-Tipe Join

Ada dua tipe utama dari kondisi join yaitu, Equijoin dan Non-equijoin. Metode tambahan join meliputi outerjoin, selfjoin, Set Operator.



5.6. Equijoin

Untuk menentukan nama pegawai dari suatu departemen, dapat membandingkan nilai di kolom DEPTNO pada tabel EMP dengan nilai DEPTNO pada tabel DEPT. Hubungan antara tabel EMP dengan tabel DEPT adalah Equijoin, maka nilai-nilai di kolom DEPTNO pada kedua table tersebut harus sama. Selain itu, EMP dan DEPT disebut Equijoin karena terdapat primary key di dalam table DEPT yang bereferensi ke foreign key di dalam table EMP.



5.7. Pengambilan Record Menggunakan Equijoin Contoh:

- Clausa SELECT mengspesifikasi nama kolom untuk diambil
  - Employee Name, Employee Number dan Department Number, dimana kolomkolom tersebut ada di tabel EMP
- Clausa FROM mengspesifikasi dua tabel database yang harus diakses

- Tabel EMP
- Tabel DEPT

- Clausa WHERE mengspesifikiasi bagaimana tabel akan join
  - EMP.DEPTNO = DEPT.DEPTNO

Kolom DEPTNO merupakan keadaan untuk kedua tabel yang harus diawali dengan nama tabel untuk menghindari kerancuan.

```
SQL> SELECT    emp.empno, emp.ename, emp.deptno,
  2            dept.deptno, dept.loc
  3 FROM       emp, dept
  4 WHERE      emp.deptno=dept.deptno;
```

Lihat Video

5.8. Menghindari Kerancuan Nama Kolom

Untuk menghindari kerancuan/ambigu maka dibutuhkan syarat dari nama kolom pada klausa WHERE dengan menyertakan nama tabel. Tanpa tabel awal, kolom DEPTNO bisa berasal dari tabel DEPT atau tabel EMP.

Jika nama kolom tidak terletak diantara kedua tabel, maka syarat dari nama kolom tidak dibutuhkan. Keperluan untuk kerancuan nama kolom yang memenuhi syarat juga digunakan untuk kolom yang mungkin rancu dalam klausa yang lain, seperti klausa SELECT atau klausa ORDER BY.

```
SQL> SELECT    emp.empno, emp.ename, emp.deptno,
  2            dept.deptno, dept.loc
  3 FROM       emp, dept
  4 WHERE      emp.deptno=dept.deptno;
```

5.9. Pencarian Kondisi Dengan Operator AND

Operator logika bisa digunakan untuk kondisi pencarian yang ada pada klausa WHERE. Contohnya, untuk menampilkan employee number, department number dan department location dari pegawai yang bernama King. Maka dibutuhkan kondisi tambahan pada klausa WHERE.

```
SQL> SELECT    emp.empno, emp.ename, emp.deptno,
  2            dept.deptno, dept.loc
  3 FROM       emp, dept
  4 WHERE      emp.deptno=dept.deptno;
```

Lihat Video

5.10. Penggunaan Tabel Alias

Persyaratan nama kolom pada nama tabel akan menghabiskan banyak waktu jika nama tabel terlalu besar. Maka, dapat digunakan alias tabel dari nama tabel. Alias tabel menyebabkan pembuat kode menjadi singkat, sehingga memori yang digunakan menjadi lebih kecil. Penggunaan tabel alias juga digunakan untuk nama kolom yang ambigu artinya nama kolom yang sama dimiliki oleh lebih dari satu tabel.

Contohnya penggunaan tabel alias, dari query yang seperti ini:

```
SQL> SELECT    emp.empno, emp.ename, emp.deptno,
  2            dept.deptno, dept.loc
  3 FROM       emp, dept
  4 WHERE      emp.deptno=dept.deptno;
```

Menjadi:

```
SQL> SELECT    e.empno, e.ename, e.deptno,
  2            d.deptno, d.loc
   3 FROM      emp e, dept d
   4 WHERE     e.deptno=d.deptno;
```

Lihat Video

Penekanan :

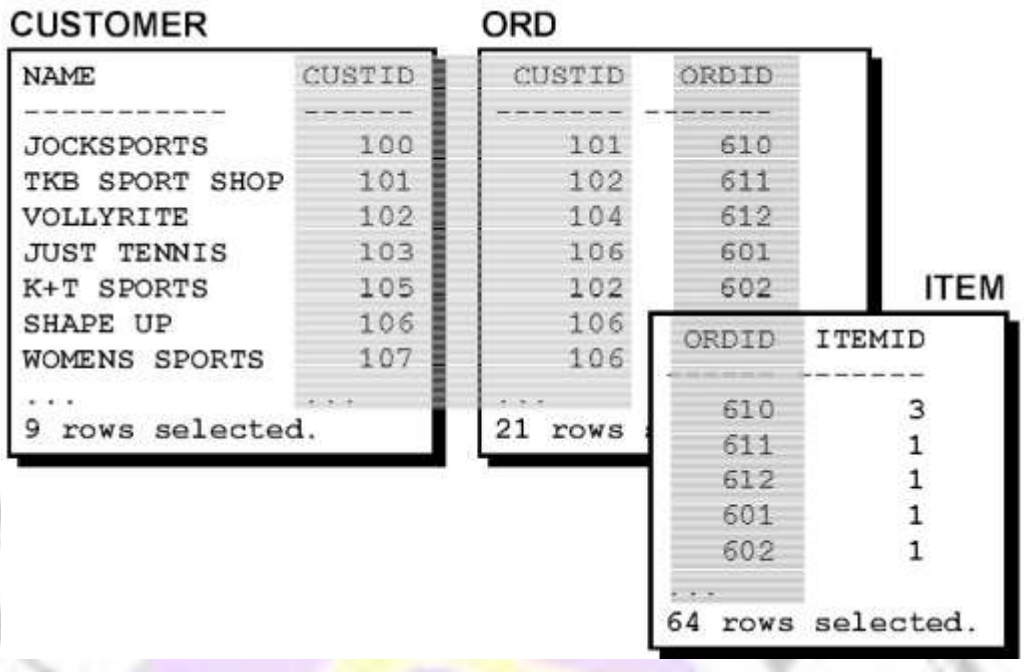
- Nama tabel alias panjang maksimum 30 karakter, tetapi lebih sedikit akan lebih baik.
- Jika tabel alias digunakan untuk nama tabel pada klausa FROM, maka tabel alias harus disubtitusi ke nama tabel melalui statement SELECT.
- Tabel alias akan sangat berarti.
- Tabel alias hanya berlaku untuk statement SELECT yang sekarang (*current*).

5.11. Join Lebih Dari Dua Tabel

Terkadang mungkin dibutuhkan join lebih dari dua tabel. Contoh JOIN lebih dari dua tabel, menampilkan *name*, *place order*, *item number*, *total* dari setiap *item*, dan total dari setiap order untuk pelanggan TKB SPORT SHOP, maka dilakukan join antara tabel CUSTOMER, tabel ORD, dan tabel ITEM.

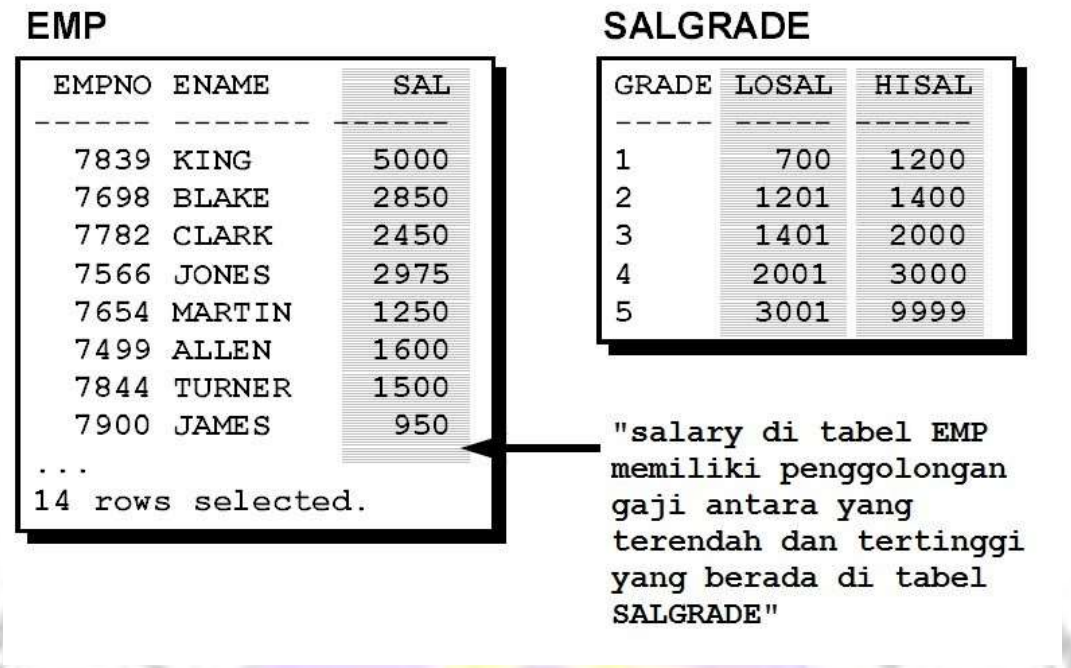
```
SQL> SELECT    c.name, o.ordid, i.itemid,
  2            i.itemtot, o.total
  3 FROM       customer c, ord o, item i
  4 WHERE      c.custid = o.custid
  5 AND        o.ordid = i.ordid
  6 AND        c.name = 'TKB SPORT SHOP';
```

Lihat Video



5.12. Non-Equijoin

Relasi antara dua tabel disebut non-equijoin jika nilai dari kolom-kolom yang dihubungkan tidak saling berhubungan secara langsung atau tidak memiliki hubungan primary key dan foreign key. Salah sati contoh dari Non-Equijoin merupakan hubungan antara tabel EMP dengan tabel SALGRADE. Hubungan antara dua tabel adalah kolom SAL dari tabel EMP dengan kolom LOSAL dan HISAL dari tabel SALGRADE. Hubungan ini meliputi penggunaan operator lain yang equal (-).



5.13. Pengambilan Record Dengan Non-Equijoin

Contoh penggunaan Non-equijoin untuk melihat evaluasi grade salary dari pegawai. Salary harus diantara range salary terendah dan tertinggi.

```
SQL> SELECT    e.ename, e.sal, s.grade,
  2  FROM      emp e, salgrade s
  3  WHERE     e.sal
  4  BETWEEN s.losal AND s.hisal;
```

Lihat Video

Catatan penting dimana semua pegawai muncul jika query dijalankan. Tidak ada pegawai yang diulang dalam tampilan. Alasannya:

- Baris kosong pada tabel grade salary yang overlap. Nilai salary untuk pegawai yang hanya terdapat antara nilai salary terendah dan salary tertinggi dari satu baris dalam tabel grade gaji.
- Seluruh salary pegawai lie dengan limit yang diinginkan pada tabel grade salary. Tidak ada pegawai dengan penghasilan lebih kecil dari nilai terendah yang terkandung dalam kolom LOSAL atau lebih besar dari nilai yang terkandung dalam kolom HISAL.

Catatan : Operator lain seperti <= dan >= dapat digunakan, tetapi BETWEEN yang sederhana. Ingat secara khusus nilai rendah pertama dan nilai tertinggi terakhir jika menggunakan BETWEEN. Tabel alias dispesifikasi untuk keadaan alasan, tanpa sebab kemungkinan rancu/ambigu.

5.14. Outerjoin

Jika terdapat baris yang tidak memenuhi kondisi join, dan akan ditampilkan pada hasil query, maka digunakan Outer Join. Operator Outerjoin adalah tanda plus (+).

Gunakan tanda plus (+) sebagai operator Outer Join, seperti ini:

```
SQL> SELECT    table1.column, table2.column
  2 FROM       table1, table2
  3 WHERE      table1.column(+) = table2.column;
```

Atau:

```
SQL> SELECT    table1.column, table2.column
  2 FROM       table1, table2
  3 WHERE      table1.column = table2.column(+);
```

Dengan :

Table1.colum adalah kondisi dimana tabel di join secara bersama.

Table2.column(+) adalah simbol outerjoin yang akan ditempatkan pada sisi lain dari kondisi

klausa WHERE, tetapi tidak untuk kedua sisi (Tempat simbol outerjoin mengikuti nama kolom dalam tabel).

5.15. Pengambilan Record Dengan Outerjoin

Gunakan outerjoin untuk melihat baris yang biasanya tidak ketemu dari kondisi join. Operator Outerjoin adalah tanda plus diapit dengan tanda kurung (+) yang ditempatkan pada sisi join yang kurang baik dalam informasi. Operator tersebut memiliki efek dari pembuatan satu atau lebih baris tabel yang baik telah di-join.

Contohnya, di dalam kondisi Equijoin pada tabel EMP dan DEPT, departemen OPERATIONS tidak muncul karena tidak ada yang bekerja di departemen sebagai OPERATIONS.

```
SQL> SELECT    e.ename, d.deptno, d.dname
  2 FROM       emp e, dept d
  3 WHERE      e.deptno(+) = d.deptno;
```

Lihat Video

Jika akan menampilkan nomor dan nama sari semua departemen dan Departemen OPERATIONS yang tidak memiliki pegawai juga dapat ditampilkan. Batasan Outerjoin:

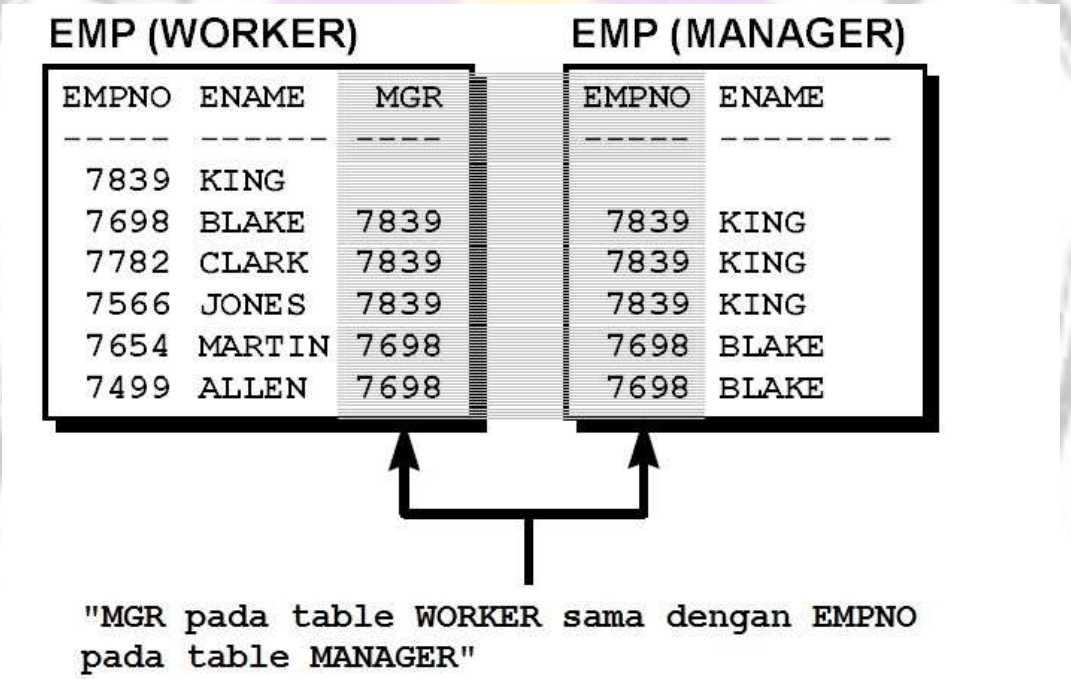
- Operator Outerjoin akan muncul hanya satu sisi dari ekspresi karena kehilangan informasi. Hal tersebut akan mengembalikan banyak baris dari satu tabel yang tidak langsung sesuai pada tabel yang lain.
- Sebuah kondisi meliputi outerjoin tidak akan menggunakan operator IN atau hubungan ke kondisi lain melalui operator OR.

```
SQL> SELECT    e.ename, d.deptno, d.dname
  2  FROM      emp e, dept d
  3  WHERE     e.deptno(+) = d.deptno
  4  ORDER BY e.deptno;
```

Lihat Video

5.16. Selfjoin

Terkadang sebuah table perlu di-joinkan dengan table itu sendiri. Untuk mencari nama dari setiap pegawai manager, maka table EMP perlu di-joinkan dengan table EMP itu sendiri.



- Mencari Blake dalam tabel EMP melalui pencarian pada kolom ENAME.
- Mencari nomor manager untuk Blake melalui pencarian pada kolom MGR. Nomor manager dari Blake adalah 7839.
- Mencari nama dari manager dengan EMPNO 7839 melalui pencari pada kolom ENAME. Nomor Pegawai King adalah 7839, maka King adalah manager dari Blake.

Prosesnya dapat dilihat pada tabel kedua. Pertama, perhatikan data dalam tabel untuk mencari Blake pada kolom ENAME dan MGR bernilai 7839. Kedua, perhatikan dalam kolom EMPNO untuk mencari 7839 dan kolom ENAME untuk mencari King.

```
SQL> SELECT    worker.ename||'work
for'||manager.ename
 2  FROM       emp worker, emp manager
 3  WHERE      worker.mgr = manager.empno;
```

Lihat Video

Contoh tersebut adalah selfjoin dari Tabel EMP. Simulasikan kedua tabel dalam klausa FROM, sehingga ada dua alias yaitu namanya WORKER dan MANAGER untuk tabel yang sama yaitu tabel EMP.

Contoh klausa WHERE berisi join yang artinya dimana nomor worker manager sesuai dengan nomor pegawai untuk manager.