




SELECT JOIN

By: Hanifah Permatasari, M.Kom

- 
- Pertemuan ini, kita akan membahas tentang query “SELECT” yang melibatkan banyak tabel nih.
 - Pusing? Tenang... query “SELECT” kali ini pakai JOIN kok. Kalau pertemuan sebelumnya, kita masih manual, jadi agak ribet karena harus nyambungin *Primary Key* dan *Foreign Key* antar tabel.

PAHAMI TABEL BERIKUT INI !

Karena tabel ini akan kita gunakan untuk praktik JOIN.

Database: MAHASISWA_NIM

Tabel mhs

nim	nama	jenis_kelamin	kd_jurusan
001	Alfa	L	J002
002	Beta	P	J002
003	Charlie	P	J001
004	Delta	L	J001
005	Erdhi	L	J001
006	Farah	P	J002
007	Gisel	P	J002
008	Haris	L	NULL

Tabel mhs_jurusan

kd_jurusan	nama_jurusan
J001	MANAJEMEN
J002	AKUNTANSI
J003	TEKNIK

**Tabel
mhs_status**

nim	status
001	LULUS
002	TIDAK LULUS
003	TIDAK LULUS
004	LULUS
005	TIDAK LULUS
006	LULUS
007	LULUS

□ Tabel mhs

nim	nama	jenis_kelamin	kd_jurusan
001	Alfa	L	J002
002	Beta	P	J002
003	Charlie	P	J001
004	Delta	L	J001
005	Erdhi	L	J001
006	Farah	P	J002
007	Gisel	P	J002
008	Haris	L	NULL

**PERHATIKAN, bahwa HARIS
(nim: 008) tidak memiliki
JURUSAN.**

□ Tabel mhs_jurusan

kd_jurusan	nama_jurusan
J001	MANAJEMEN
J002	AKUNTANSI
J003	TEKNIK

**PERHATIKAN, bahwa
TEKNIK(kode: J003) tidak diambil
oleh mahasiswa siapapun.**

nim	status
001	LULUS
002	TIDAK LULUS
003	TIDAK LULUS
004	LULUS
005	TIDAK LULUS
006	LULUS
007	LULUS

□ Tabel mhs_status

**PERHATIKAN,
bahwa HARIS
(NIM: 008) tidak
ada statusnya.**



Now, kita masuk ke materi JOIN

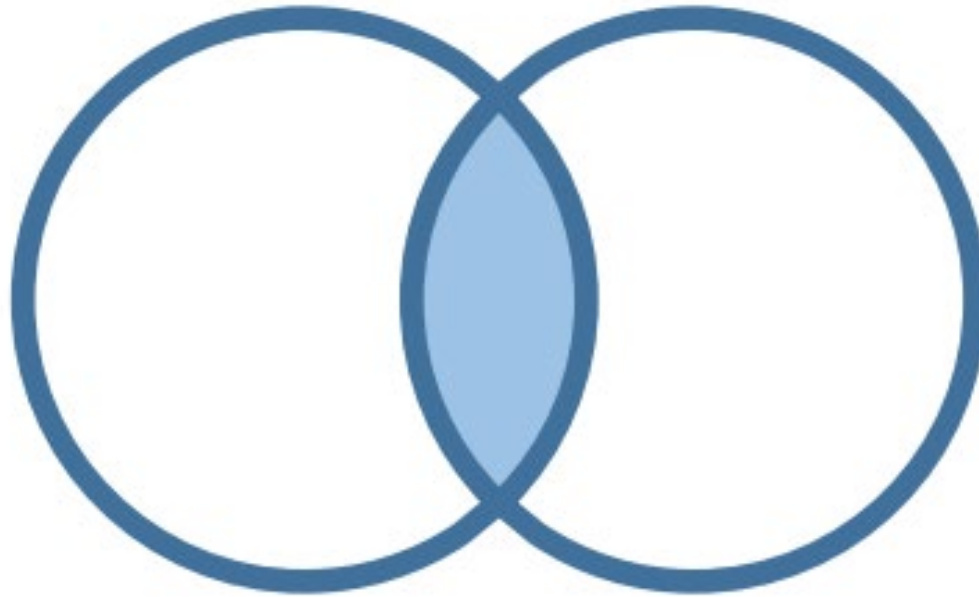
Jenis JOIN

- ❑ Inner Join atau Cross Join
- ❑ Outer Join
 - ▣ OUTER JOIN ini dibagi menjadi tiga yaitu LEFT OUTER JOIN, RIGHT OUTER JOIN, dan FULL OUTER JOIN.
 - ▣ Namun, sampai dengan saat ini, MySQL hanya *men-support* LEFT OUTER JOIN dan RIGHT OUTER JOIN.



Pertama, INNER JOIN

- Pada inner/cross join, data yang yang ditampilkan hanya data yang ada di **kedua tabel**.



- Misalnya, kita ingin menampilkan data **mahasiswa** lengkap dengan **jurusannya**.

nim	nama	Jenis_kelamin	Kd_jurusan	Nama_jurusan
001	Alfa	L	J002	Manajemen
Dst...				

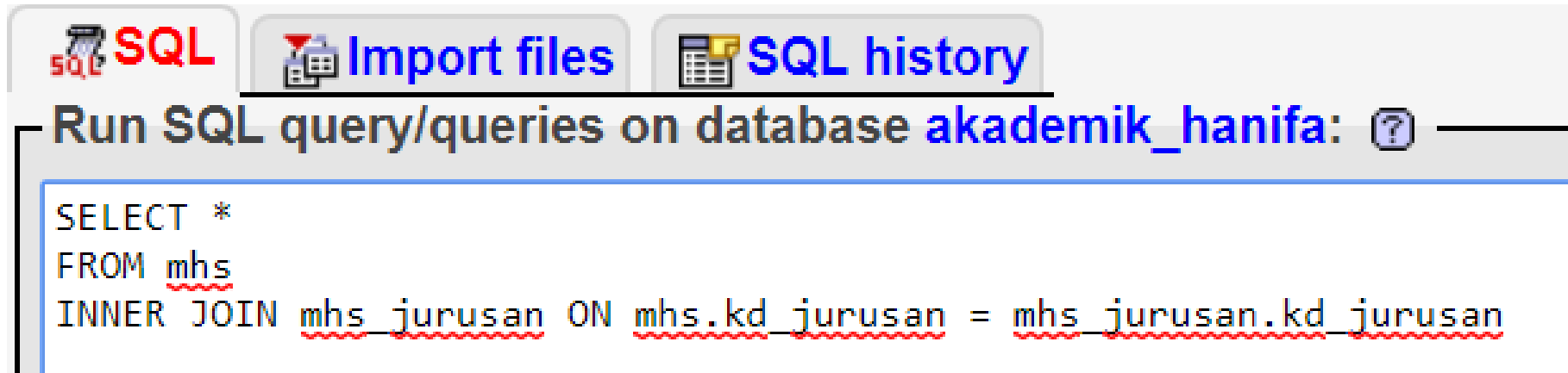
- Pertama, JOIN pakai USING :

Run SQL query/queries on database **akademik_hanifa**: (?) —

```
SELECT *  
FROM mhs  
JOIN mhs_jurusan USING (kd_jurusan)|
```

Pada query di atas, tabel **mhs** akan direlasikan dengan tabel **mhs_jurusan**. Kedua tabel ini akan dihubungkan dengan kolom **kd_jurusan**

- Selain USING, kita juga dapat menggunakan keyword ON.



The screenshot shows a SQL interface with three tabs: "SQL", "Import files", and "SQL history". The "SQL" tab is active, and the text "Run SQL query/queries on database akademik_hanifa:" is displayed. Below this, a SQL query is entered in a text area:

```
SELECT *  
FROM mhs  
INNER JOIN mhs_jurusan ON mhs.kd_jurusan = mhs_jurusan.kd_jurusan
```

The underlines in the query are red, and the "ON" keyword is highlighted in blue.

Pada query di atas, sama seperti query SELECT pada pertemuan sebelumnya. Ribet ya, karena harus menghubungkan kolom primary key dan foreign key nya. Kalau Ribet, kita pakai cara “USING” aja ya

CATATAN PENTING!

- ❑ Klausa INNER JOIN dapat diganti dengan CROSS JOIN, atau cukup dengan JOIN saja, ketiganya akan menghasilkan output yang sama.
- ❑ Pada sesion ini, kita akan selalu menggunakan klausa **JOIN** tanpa tambahan INNER atau CROSS.

***SETELAH QUERY DI EKSEKUSI,
KOK “HARIS” NGGA ADA,
BU?***

***JURUSAN “TEKNIK” JUGA
NGGA ADA, BU?***

- Pada contoh tersebut, kita menggabungkan data mahasiswa dengan jurusan menggunakan kolom kode_jurusan.
- Dari tabel yang dihasilkan, terlihat bahwa mahasiswa dengan nama Haris tidak ditampilkan, karena kode jurusan nya yaitu **NULL** (tidak ada pada tabel mhs_jurusan).
- Demikian juga dengan Jurusan teknik, jurusan ini tidak ditampilkan karena kode jurusan J003 tidak ada pada tabel mhs

*Jadi intinya, query **JOIN** ini cuma mau nampilin data yang konsisten aja. Kalau di tabel A si Haris ada, tapi di tabel B si Haris gak ada, maka **JOIN** nggak mau nampilin data Haris sama sekali.*

*Kalau kamu ada di hatinya, dan dia ada dihatimu, **JOIN** mau nampilin data kamu dan dia.*

*Tapi kalau dia ada dihatimu, dan kamu **NGGAK** ada dihatinya, **JOIN** nggak mau nampilin data kamu dan dia. Kasihan...*



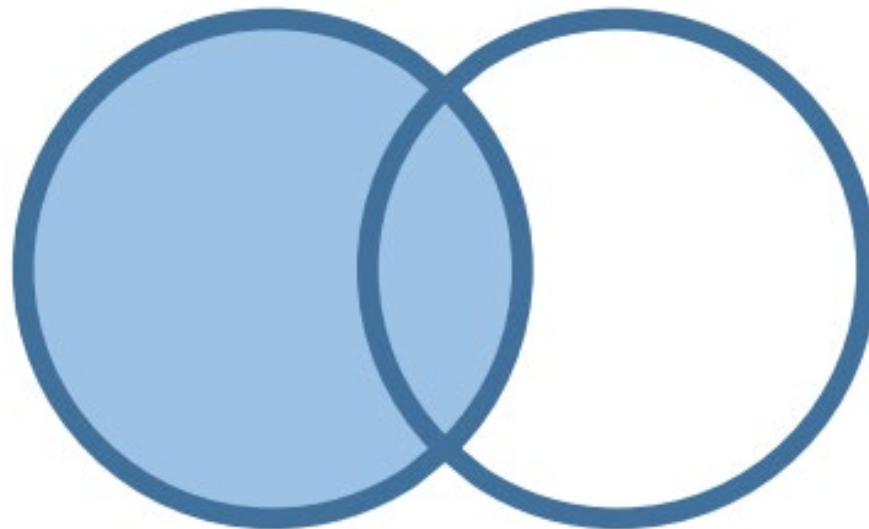
Kedua, OUTER JOIN

Outer Join

- ❑ Seperti namanya, OUTER JOIN akan menampilkan semua data pada salah satu tabel dan data pada tabel lain yang terhubung dengan tabel pertama.
- ❑ OUTER JOIN ini dibagi menjadi tiga yaitu LEFT OUTER JOIN, RIGHT OUTER JOIN, dan FULL OUTER JOIN.
- ❑ Namun, sampai dengan saat ini, MySQL hanya *men-support* LEFT OUTER JOIN dan RIGHT OUTER JOIN.

1. LEFT OUTER JOIN atau LEFT JOIN

- ❑ LEFT OUTER JOIN akan menampilkan semua data pada tabel sebelah kiri dan data yang ada di tabel sebelah kanan, khusus yang datanya ada pada tabel sebelah kiri.



NOTE!

- Jika data pada tabel sebelah kiri tidak memiliki pasangan data dengan tabel di sebelah kanan, maka data pada tabel sebelah kanan akan bernilai NULL

MARI KITA EKSEKUSI KEMBALI

Data mahasiswa dengan data jurusan mahasiswa

Run SQL query/queries on database **akademik_hanifa**:

```
SELECT *  
FROM mhs  
LEFT JOIN mhs_jurusan USING (kd_jurusan)
```

<u>kd_jurusan</u>	<u>nim</u>	<u>nama</u>	<u>jenis_kelamin</u>	<u>nama_jurusan</u>
J002	001	Alfa	L	AKUNTANSI
J002	002	Beta	P	AKUNTANSI
J001	003	Charlie	P	MANAJEMEN
J001	004	Delta	L	MANAJEMEN
J001	005	Erdhi	L	MANAJEMEN
J002	006	Farah	P	AKUNTANSI
J002	007	Gisel	P	AKUNTANSI
NULL	008	Haris	L	NULL

- Contoh di atas, data mahasiswa bernama Haris tetap tampil, dan tidak menampilkan data jurusan teknik.
WHY?

Penjelasan Posisi Kanan/Kiri

- ❑ `SELECT * FROM mhs`
- ❑ `LEFT JOIN mhs_jurusan USING (kd_jurusan)`

Tabel yang KIRI adalah **mhs**

Tabel yang KANAN adalah **mhs_jurusan**

- Karena seperti yang telah disebutkan sebelumnya, pada LEFT JOIN data pada **tabel disebelah kiri**, yaitu **tabel mhs** akan ditampilkan semua, sedangkan data dengan kode jurusan J003 yang ada pada **tabel mhs_jurusan (tabel sebelah kanan)** tidak ditampilkan karena tidak memiliki pasangan dengan data pada tabel mhs.
- Perhatikan bahwa : karena Haris tidak memiliki data pada kolom kd_jurusan, maka kolom nama_jurusan (kolom tabel sebelah kanan) untuk Haris bernilai NULL

- ❑ Kamu di kiri, dia di kanan. Dihatimu ada dia, tapi hatinya dia nggak ada kamu. Hatimu gak punya pasangan deh.
- ❑ Maka, sistem akan menampilkan kamu, tapi NULL untuk hatinya dia.

HATI KAMU #Kiri
(Ada dia)



HATI DIA#Kanan
(Kamu Ngga Ada)

Tampilan Sistem :

HATI KAMU

NULL

2. RIGHT OUTER JOIN atau RIGHT JOIN

- Kebalikan dari left join, pada right join, semua data pada tabel disebelah kanan akan ditampilkan semua sedangkan data pada tabel disebelah kiri ditampilkan hanya jika data tersebut ada pada tabel disebelah kanan.
- Jika data pada tabel sebelah kanan tidak memiliki pasangan dengan tabel di sebelah kiri, maka data pada tabel sebelah kiri akan bernilai NULL

MARI KITA EKSEKUSI KEMBALI

Data mahasiswa dengan data jurusan mahasiswa

Run SQL query/queries on database **akademik_hanifa**: 

```
SELECT *  
FROM mhs  
RIGHT JOIN mhs_jurusan USING (kd_jurusan)  
|
```

kd_jurusan	nama_jurusan	nim	nama	jenis_kelamin
J001	MANAJEMEN	003	Charlie	P
J001	MANAJEMEN	004	Delta	L
J001	MANAJEMEN	005	Erdhi	L
J002	AKUNTANSI	001	Alfa	L
J002	AKUNTANSI	002	Beta	P
J002	AKUNTANSI	006	Farah	P
J002	AKUNTANSI	007	Gisel	P
J003	TEKNIK	NULL	NULL	NULL

HATI KAMU #Kiri
(Ada dia)



HATI DIA#Kanan
(Kamu Ngga Ada)

Tampilan Sistem :

NULL

HATI KAMU

Ini mau dibolak balik pake left join atau right join, HATI DIA tetep tampil NULL, anak-anak. Karena kalau isinya tetap, mau pindah posisi gimanapun ya tetep hasilnya begitu.

Sabar ya



Tips: kita cukup menghafal JOIN dan LEFT JOIN saja. Kenapa? Karena RIGHT JOIN ini secara mudah dapat diganti dengan LEFT JOIN yaitu hanya dengan mengubah posisi tabel nya, misal pada contoh diatas kita ubah posisi tabel menjadi **FROM mhs_jurusan LEFT JOIN mhs**, hasil yang kita peroleh akan sama, sehingga untuk mempermudah pemahaman, sebaiknya cukup menggunakan LEFT JOIN saja.

3. NATURAL JOIN

- ❑ Sejah ini, ketika kita melakukan join tabel, kita harus mendefinisikan bentuk join (INNER atau OUTER JOIN) dan hubungan antar tabel yang digabungkan menggunakan klausa ON atau USING.
- ❑ Nah, ternyata terdapat cara yang lebih mudah, yaitu cukup mendefinisikan bentuk JOIN-nya saja, tanpa mendefinisikan hubungan antar tabel.
- ❑ Caranya, tambahkan klausa NATURAL sebelum klausa bentuk join.

Run SQL query/queries on database **akademik_hanifa**: ?

```
SELECT *  
FROM mhs  
NATURAL JOIN mhs_jurusan
```

kd_jurusan	nim	nama	jenis_kelamin	nama_jurusan
J002	001	Alfa	L	AKUNTANSI
J002	002	Beta	P	AKUNTANSI
J001	003	Charlie	P	MANAJEMEN
J001	004	Delta	L	MANAJEMEN
J001	005	Erdhi	L	MANAJEMEN
J002	006	Farah	P	AKUNTANSI
J002	007	Gisel	P	AKUNTANSI

Tips & Trick

- ❑ Kalau pengen data yang gak punya pasangan tidak ditampilkan sama sekali, ya pakai NATURAL JOIN.
- ❑ Kalau pengen data yang gak punya pasangan di data sebelahnya tetap ditampilin meskipun keluar identitas NULL, ya pakai NATURAL LEFT JOIN, atau NATURAL RIGHT JOIN

MARI EKSEKUSI 3 TABEL

Tabel mhs, mhs_jurusan, dan mhs_status

Run SQL query/queries on database **akademik_hanifa**:

```
SELECT nim, nama, jur.nama_jurusan AS jurusan_mahasiswa, st.status AS status_mahasiswa  
FROM mhs  
LEFT JOIN mhs_jurusan AS jur USING (kd_jurusan)  
LEFT JOIN mhs_status AS st USING (nim)
```

nim	nama	jurusan_mahasiswa	status_mahasiswa
001	Alfa	AKUNTANSI	LULUS
002	Beta	AKUNTANSI	TIDAK LULUS
003	Charlie	MANAJEMEN	TIDAK LULUS
004	Delta	MANAJEMEN	LULUS
005	Erdhi	MANAJEMEN	TIDAK LULUS
006	Farah	AKUNTANSI	LULUS
007	Gisel	AKUNTANSI	LULUS
008	Haris	NULL	NULL

Itu kok query nya jadi panjang banget Bu?

```
SELECT nim, nama, jur.nama_jurusan AS jurusan_mahasiswa, st.status AS status_mahasiswa  
FROM mhs  
LEFT JOIN mhs_jurusan AS jur USING (kd_jurusan)  
LEFT JOIN mhs_status AS st USING (nim)
```

- ❑ Pertama, kita bikin *alias* untuk tabel **mhs_jurusan** dan tabel **mhs_status**.
- ❑ **jur** adalah alias tabel **mhs_jurusan** , **st** adalah alias tabel **mhs_status**.
- ❑ Kedua, kita bikin alias untuk kolom **nama_jurusan**, dan kolom **status**.
- ❑ **Jurusan_mahasiswa** adalah alias kolom **nama_jurusan**, **status_mahasiswa** adalah alias kolom **status**.
- ❑ Nama alias itu terletak disebelah sintak **AS**.

- ❑ Versi simpelnya begini :

```
SELECT nim, nama, nama_jurusan, status  
FROM mhs  
LEFT JOIN mhs_jurusan USING (kd_jurusan)  
LEFT JOIN mhs_status USING (nim)
```

- ❑ Tapi nanti hasil tampilannya begini :

NIM	NAMA	NAMA_JURUSAN	STATUS
-----	------	--------------	--------

- ❑ Kalau pengen begini, pake query yang ada ALIAS nya tadi

NIM	NAMA	JURUSAN_MAHASISWA	STATUS_MAHASISWA
-----	------	-------------------	------------------



Gimana kalau pake Natural JOIN?

Run SQL query/queries on database akademik_hanifa:

```
SELECT nim, nama, jur.nama_jurusan AS jurusan_mahasiswa, st.status AS  
status_mahasiswa  
FROM mhs  
NATURAL JOIN mhs_jurusan jur  
NATURAL JOIN mhs_status st
```

nim	nama	jurusan_mahasiswa	status_mahasiswa
001	Alfa	AKUNTANSI	LULUS
002	Beta	AKUNTANSI	TIDAK LULUS
003	Charlie	MANAJEMEN	TIDAK LULUS
004	Delta	MANAJEMEN	LULUS
005	Erdhi	MANAJEMEN	TIDAK LULUS
006	Farah	AKUNTANSI	LULUS
007	Gisel	AKUNTANSI	LULUS

PERHATIKAN !

- ❑ Hati hati ketika menggunakan NATURAL, pastikan bahwa **nama kolom yang sama** benar benar menjadi penghubung kedua tabel, jika tidak, maka kita akan memperoleh hasil yang tidak diharapkan.
- ❑ Jadi kalau mau pakai NATURAL, primary dan foreign yang jadi penghubung, namanya harus sama ya. Kalau di tabel mahasiswa pake NIM, ya di tabel status mahasiswa pake NIM juga. Jangan satu pakai NIM, yang satu pakai Nomor Mahasiswa.



JOIN juga bisa pakai WHERE lho!

- Misalkan kita pengen menampilkan data mahasiswa yang statusnya lulus aja. Berarti?
- Habis join, tambahi WHERE status='LULUS'
- Kalau yang statusnya lulus dan jurusanannya akuntansi?
- Ya tambahi WHERE status='LULUS' AND nama_jurusan='AKUNTANSI'.

NEXT, Mari Kita Latihan!

Pada pertemuan kemarin, Anda kan merelasikan tabel secara MANUAL.

Nah, dengan **SOAL YANG SAMA**, relasikan tabel menggunakan JOIN/LEFT JOIN/NATURAL, terserah enaknya kamu mau pakai yang mana.



COMING SOON, NEXT WEEK!

Kita akan mulai membahas percabangan (IF/CASE),
dan agregasi (SUM/COUNT/AVG/dsb).

Kencangkan ikat kepala, *and stay healthy.*