

# BASIS DATA

Join Query, Query Optimization & Tuning

# Query **JOIN** di SQL

**Tujuan:** Menggabungkan data dari beberapa tabel.

Selain **JOIN**, penggabungan juga bisa dilakukan dengan **WHERE**.

Jenis-jenis **JOIN**:

- **INNER JOIN** – menampilkan data yang cocok di kedua tabel.
- **LEFT JOIN** – menampilkan semua data dari tabel kiri.
- **RIGHT JOIN** – menampilkan semua data dari tabel kanan.

```
SELECT nama_kolom FROM tabel1 JOIN tabel2  
WHERE tabel1.kolom1 = tabel2.kolom1
```

# Membuat tabel universitas

```
-- Membuat tipe ENUM untuk akreditasi
CREATE TYPE akreditasi_enum AS ENUM ('A', 'B', 'C', 'N/A');

-- Membuat tabel universitas
CREATE TABLE universitas (
    jurusan VARCHAR(20),
    tgl_berdiri DATE,
    nama_dekan VARCHAR(50),
    jum_mhs SMALLINT CHECK (jum_mhs >= 0), -- menggantikan UNSIGNED
    akr akreditasi_enum
);

INSERT INTO universitas (jurusan, tgl_berdiri, nama_dekan, jum_mhs, akr) VALUES
('Kimia', '1987-07-12', 'Prof. Mulyono, M.Sc.', 662, 'B'),
('Ilmu Komputer', '2003-02-23', 'Dr. Syahrial, M.Kom.', 412, 'A'),
('Akuntansi', '1985-03-19', 'Maya Fitrianti, M.M.', 895, 'B'),
('Farmasi', '1997-05-30', 'Prof. Silvia Nst, M.Farm.', 312, 'C'),
('Fisika', '1989-12-10', 'Dr. Umar Agustinus, M.Sc.', 275, 'A'),
('Hukum', '1983-08-08', 'Prof. Gunarto, M.H.', 754, 'B');
```

```
-- Membuat tipe ENUM untuk jenis kelamin
CREATE TYPE jenis_kelamin_enum AS ENUM ('L', 'P');

-- Membuat tabel mahasiswa
CREATE TABLE mahasiswa (
    nama VARCHAR(50),
    asal VARCHAR(50),
    kel jenis_kelamin_enum,
    tinggi SMALLINT CHECK (tinggi >= 0),
    jurusan VARCHAR(20),
    nilai_uan NUMERIC(5,2)
);

-- Insert data
INSERT INTO mahasiswa (nama, asal, kel, tinggi, jurusan, nilai_uan) VALUES
('Riana Putra', 'Padang', 'P', 155, 'Kimia', 339.20),
('Rudi Permana', 'Bandung', 'L', 163, 'Ilmu Komputer', 290.44),
('Sari Citra Lestari', 'Jakarta', 'P', 161, 'Manajemen', 310.60),
('Rina Kumala Sari', 'Jakarta', 'P', 158, 'Akuntansi', 337.99),
('James Situmorang', 'Medan', 'L', 168, 'Kedokteran Gigi', 341.10),
('Sandri Fatmala', 'Bandung', 'P', 165, 'Ilmu Komputer', 322.91),
('Husli Khairan', 'Jakarta', 'L', 170, 'Akuntansi', 288.55),
('Christine Wijaya', 'Medan', 'P', 157, 'Manajemen', 321.74),
('Ikhsan Prayoga', 'Jakarta', 'L', 172, 'Ilmu Komputer', 300.16),
('Bobby Permana', 'Medan', 'L', 161, 'Ilmu Komputer', 280.82);
```

SELECT \* FROM universitas LIMIT 100

jurusan	tgl_berdiri	nama_dekan	jum_mhs	akr
varchar(20)	date	varchar(50)	smallint	akreditasi_enum
> Kimia	1987-07-12	Prof. Mulyono, M.Sc.	662	B
> Ilmu Komputer	2003-02-23	Dr. Syahrial, M.Kom.	412	A
> Akuntansi	1985-03-19	Maya Fitrianti, M.M.	895	B
> Farmasi	1997-05-30	Prof. Silvia Nst, M.Farm.	312	C
> Fisika	1989-12-10	Dr. Umar Agustinus, M.Sc.	275	A
> Hukum	1983-08-08	Prof. Gunarto, M.H.	754	B

Tidak semua jurusan dari tabel mahasiswa ada di tabel universitas dan begitu juga sebaliknya.

Di tabel mahasiswa, terdapat jurusan Kedokteran Gigi dan Manajemen. Jurusan ini tidak ada di tabel universitas. Sedangkan di tabel universitas terdapat jurusan Farmasi, Fisika dan Hukum. Ketiga jurusan ini juga tidak dipilih mahasiswa di tabel mahasiswa. Hubungan ini merupakan kunci kita agar bisa memahami berbagai query JOIN.

SELECT \* FROM mahasiswa LIMIT 100

nama	asal	kel	tinggi	jurusan	nilai_uan
varchar(50)	varchar(50)	jenis_kelamin_enum	smallint	varchar(20)	numeric(5,2)
> Riana Putra	Padang	P	155	Kimia	339.20
> Rudi Permana	Bandung	L	163	Ilmu Komputer	290.44
> Sari Citra Lestari	Jakarta	P	161	Manajemen	310.60
> Rina Kumala Sari	Jakarta	P	158	Akuntansi	337.99
> James Situmorang	Medan	L	168	Kedokteran Gigi	341.10
> Sandri Fatmala	Bandung	P	165	Ilmu Komputer	322.91
> Husli Khairan	Jakarta	L	170	Akuntansi	288.55
> Christine Wijaya	Medan	P	157	Manajemen	321.74
> Ikhsan Prayoga	Jakarta	L	172	Ilmu Komputer	300.16
> Bobby Permana	Medan	L	161	Ilmu Komputer	280.82

```
SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan  
FROM mahasiswa, universitas  
WHERE mahasiswa.jurusan = universitas.jurusan;
```

Run | +Tab | JSON | Ask AI

```
24 SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan  
25 FROM mahasiswa, universitas  
26 WHERE mahasiswa.jurusan = universitas.jurusan; 7ms
```

Result(RO) X

Search Results

	nama	jurusan	nama_dekan
>	Riana Putra	Kimia	Prof. Mulyono, M.Sc.
>	Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
>	Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.



```
SELECT * FROM mahasiswa;
```

nama	asal	kel	tinggi	jurusan	nilai_uan
Riana Putra	Padang	P	155	Kimia	339.20
Rudi Permana	Bandung	L	163	Ilmu Komputer	290.44
Sari Citra Lestari	Jakarta	P	161	Manajemen	310.60
Rina Kumala Sari	Jakarta	P	158	Akuntansi	337.99
James Situmorang	Medan	L	168	Kedokteran Gigi	341.10
Sandri Fatmala	Bandung	P	165	Ilmu Komputer	322.91
Husli Khairan	Jakarta	L	170	Akuntansi	288.55
Christine Wijaya	Medan	P	157	Manajemen	321.74
Ikhsan Prayoga	Jakarta	L	172	Ilmu Komputer	300.16
Bobby Permana	Medan	L	161	Ilmu Komputer	280.82

```
SELECT * FROM universitas;
```

jurusan	tgl_berdiri	nama_dekan	jum_mhs	akr
Kimia	1987-07-12	Prof. Mulyono, M.Sc.	662	B
Ilmu Komputer	2003-02-23	Dr. Syahrial, M.Kom.	412	A
Akuntansi	1985-03-19	Maya Fitrianti, M.M.	895	B
Farmasi	1997-05-30	Prof. Silvia Nst, M.Farm.	312	C
Fisika	1989-12-10	Dr. Umar Agustinus, M.Sc.	275	A
Hukum	1983-08-08	Prof. Gunarto, M.H.	754	B

```
SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan
FROM mahasiswa, universitas
WHERE mahasiswa.jurusan = universitas.jurusan;
```

nama	jurusan	nama_dekan
Riana Putra	Kimia	Prof. Mulyono, M.Sc.
Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.
Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.

Prinsip yang sama juga dipakai untuk query INNER JOIN.  
Perintah diatas bisa dikonversi menjadi query berikut ini:

```
Run | +Tab | JSON | Select | Ask AI
28 SELECT m.nama, m.jurusan, u.nama_dekan
29 FROM mahasiswa m
30 INNER JOIN universitas u
31 ON m.jurusan = u.jurusan; 2ms
32
```

Result(RO) X

Search Results

nama	jurusan	nama_dekan
> Riana Putra	Kimia	Prof. Mulyono, M.Sc.
> Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
> Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
> Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
> Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
> Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
> Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.

Prinsip utama dari INNER JOIN adalah nilai dari kolom jurusan harus tersedia di kedua tabel. Di lain pihak, query LEFT JOIN dan RIGHT JOIN akan memaksa salah satu tabel untuk tetap ditampilkan nilainya (meskipun tidak memiliki pasangan).



```
Run | +Tab | JSON | Select | Ask AI
34 SELECT m.nama, m.jurusan, u.nama_dekan
35 FROM mahasiswa m
36 LEFT JOIN universitas u
37 ON m.jurusan = u.jurusan; 5ms
38
```

Result(RO) X

Search Results

	nama	jurusan	nama_dekan
>	Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
>	Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
>	Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.
>	Christine Wijaya	Manajemen	(NULL)
>	Sari Citra Lestari	Manajemen	(NULL)
>	James Situmorang	Kedokteran Gigi	(NULL)

Dengan query LEFT JOIN, seluruh nama mahasiswa akan ditampilkan, termasuk yang memiliki jurusan Manajemen dan Kedokteran Gigi. Informasi mengenai nama dekan untuk kedua jurusan ini tidak tersedia di tabel universitas, karena itulah nilainya diganti menjadi NULL.

```
SELECT * FROM mahasiswa;
```

nama	asal	kel	tinggi	jurusan	nilai_uan
Riana Putria	Padang	P	155	Kimia	339.20
Rudi Permana	Bandung	L	163	Ilmu Komputer	290.44
Sari Citra Lestari	Jakarta	P	161	Manajemen	310.60
Rina Kumala Sari	Jakarta	P	158	Akuntansi	337.99
James Situmorang	Medan	L	168	Kedokteran Gigi	341.10
Sandri Fatmala	Bandung	P	165	Ilmu Komputer	322.91
Husli Khairan	Jakarta	L	170	Akuntansi	288.55
Christine Wijaya	Medan	P	157	Manajemen	321.74
Ikhsan Prayoga	Jakarta	L	172	Ilmu Komputer	300.16
Bobby Permana	Medan	L	161	Ilmu Komputer	280.82

```
SELECT * FROM universitas;
```

jurusan	tgl_berdiri	nama_dekan	jum_mhs	akr
Kimia	1987-07-12	Prof. Mulyono, M.Sc.	662	B
Ilmu Komputer	2003-02-23	Dr. Syahrial, M.Kom.	412	A
Akuntansi	1985-03-19	Maya Fitrianti, M.M.	895	B
Farmasi	1997-05-30	Prof. Silvia Nst, M.Farm.	312	C
Fisika	1989-12-10	Dr. Umar Agustinus, M.Sc.	275	A
Hukum	1983-08-08	Prof. Gunarto, M.H.	754	B

```
SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan
FROM mahasiswa LEFT JOIN universitas
ON mahasiswa.jurusan = universitas.jurusan;
```

nama	jurusan	nama_dekan
Riana Putria	Kimia	Prof. Mulyono, M.Sc.
Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.
Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
Sari Citra Lestari	Manajemen	NULL
James Situmorang	Kedokteran Gigi	NULL
Christine Wijaya	Manajemen	NULL

Query LEFT JOIN, juga berarti: “Ambil seluruh data dari tabel sebelah kiri, dan tampilkan nilainya meskipun tidak berpasangan”. Tabel sebelah kiri ini adalah tabel mahasiswa, karena pada saat penulisan query, tabel mahasiswa-lah yang saya tempatkan di sisi kiri: FROM mahasiswa LEFT JOIN universitas

Sebaliknya, query RIGHT JOIN akan menampilkan seluruh tabel di sisi kanan, yakni tabel universitas, Meskipun tidak ada mahasiswa yang memilih jurusan tersebut:

Run | +Tab | JSON | Select | Ask AI

39 SELECT m.nama, m.jurusan, u.nama\_dekan

40 FROM mahasiswa m

41 RIGHT JOIN universitas u

42 ON m.jurusan = u.jurusan; 3ms

43


44

Result(RO) X

Search Results

Export

nama	jurusan	nama_dekan
Riana Putra	Kimia	Prof. Mulyono, M.Sc.
Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.
(NULL)	(NULL)	Prof. Silvia Nst, M.Farm.
(NULL)	(NULL)	Dr. Umar Agustinus, M.Sc.
(NULL)	(NULL)	Prof. Gunarto, M.H.



Administrasi Basis Data

Halaman :



```
SELECT * FROM mahasiswa;
```

nama	asal	kel	tinggi	jurusan	nilai_uan
Riana Putra	Padang	P	155	Kimia	339.20
Rudi Permana	Bandung	L	163	Ilmu Komputer	290.44
Sari Citra Lestari	Jakarta	P	161	Manajemen	310.60
Rina Kumala Sari	Jakarta	P	158	Akuntansi	337.99
James Situmorang	Medan	L	168	Kedokteran Gigi	341.10
Sandri Fatmala	Bandung	P	165	Ilmu Komputer	322.91
Husli Khairan	Jakarta	L	170	Akuntansi	288.55
Christine Wijaya	Medan	P	157	Manajemen	321.74
Ikhsan Prayoga	Jakarta	L	172	Ilmu Komputer	300.16
Bobby Permana	Medan	L	161	Ilmu Komputer	280.82

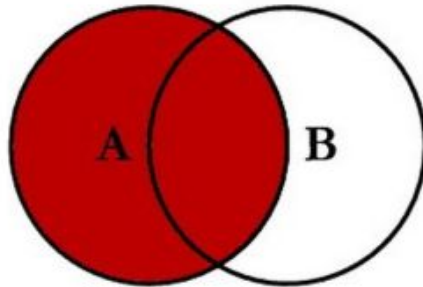
```
SELECT * FROM universitas;
```

jurusan	tgl_berdiri	nama_dekan	jum_mhs	akr
Kimia	1987-07-12	Prof. Mulyono, M.Sc.	662	B
Ilmu Komputer	2003-02-23	Dr. Syahrial, M.Kom.	412	A
Akuntansi	1985-03-19	Maya Fitrianti, M.M.	895	B
Farmasi	1997-05-30	Prof. Silvia Nst, M.Farm.	312	C
Fisika	1989-12-10	Dr. Umar Agustinus, M.Sc.	275	A
Hukum	1983-08-08	Prof. Gunarto, M.H.	754	B

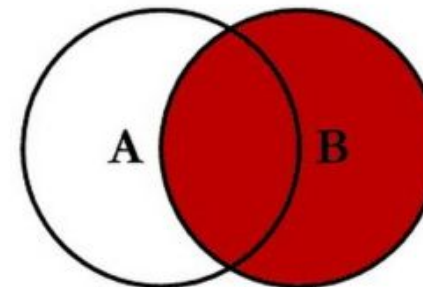
```
SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan
FROM mahasiswa RIGHT JOIN universitas
ON mahasiswa.jurusan = universitas.jurusan;
```

nama	jurusan	nama_dekan
Riana Putra	Kimia	Prof. Mulyono, M.Sc.
Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.
Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
NULL	NULL	Prof. Silvia Nst, M.Farm.
NULL	NULL	Dr. Umar Agustinus, M.Sc.
NULL	NULL	Prof. Gunarto, M.H.

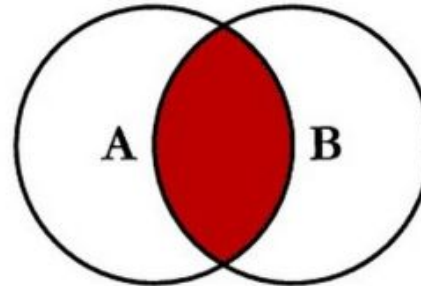
# SQL JOINS



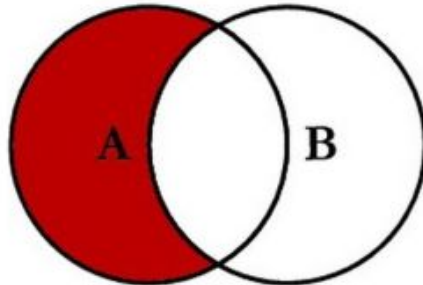
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



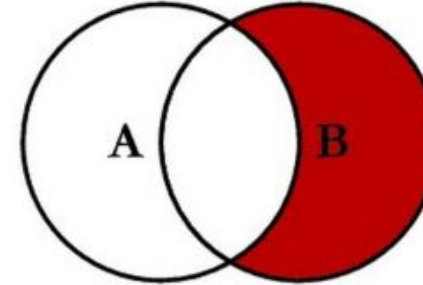
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



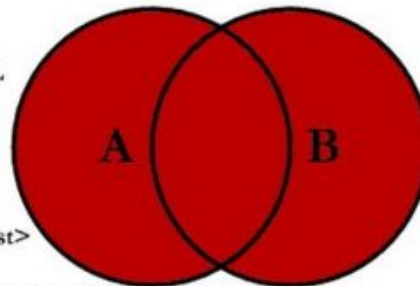
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



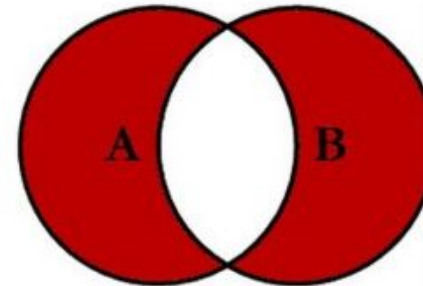
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

© C.L. Moffatt, 2008

```
mysql> SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan  
-> FROM mahasiswa LEFT JOIN universitas  
-> ON mahasiswa.jurusan = universitas.jurusan WHERE universitas.jurusan IS NULL;
```

nama	jurusan	nama_dekan
Sari Citra Lestari	Manajemen	NULL
James Situmorang	Kedokteran Gigi	NULL
Christine Wijaya	Manajemen	NULL

3 rows in set (0,00 sec)

```
mysql> SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan  
-> FROM mahasiswa RIGHT JOIN universitas  
-> ON mahasiswa.jurusan = universitas.jurusan WHERE mahasiswa.jurusan IS NULL;
```

nama	jurusan	nama_dekan
NULL	NULL	Prof. Silvia Nst, M.Farm.
NULL	NULL	Dr. Umar Agustinus, M.Sc.
NULL	NULL	Prof. Gunarto, M.H.

3 rows in set (0,00 sec)



```
mysql> SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan
-> FROM mahasiswa LEFT JOIN universitas
-> ON mahasiswa.jurusan = universitas.jurusan
-> UNION
-> SELECT mahasiswa.nama, mahasiswa.jurusan, universitas.nama_dekan
-> FROM mahasiswa RIGHT JOIN universitas
-> ON mahasiswa.jurusan = universitas.jurusan;
```

nama	jurusan	nama_dekan
Riana Putra	Kimia	Prof. Mulyono, M.Sc.
Rudi Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
Sari Citra Lestari	Manajemen	NULL
Rina Kumala Sari	Akuntansi	Maya Fitrianti, M.M.
James Situmorang	Kedokteran Gigi	NULL
Sandri Fatmala	Ilmu Komputer	Dr. Syahrial, M.Kom.
Husli Khairan	Akuntansi	Maya Fitrianti, M.M.
Christine Wijaya	Manajemen	NULL
Ikhsan Prayoga	Ilmu Komputer	Dr. Syahrial, M.Kom.
Bobby Permana	Ilmu Komputer	Dr. Syahrial, M.Kom.
NULL	NULL	Prof. Silvia Nst, M.Farm.
NULL	NULL	Dr. Umar Agustinus, M.Sc.
NULL	NULL	Prof. Gunarto, M.H.

13 rows in set (0,02 sec)

## Apa Itu Query?

- Query = perintah untuk meminta data dari database
- Contoh: `SELECT nama, jurusan FROM mahasiswa WHERE jurusan = 'Informatika';`
- `SELECT` → pilih kolom
- `FROM` → tabel sumber
- `WHERE` → filter data
- Contoh: menampilkan semua mahasiswa jurusan Informatika.

# Bagaimana Database Menjalankan Query?

- PostgreSQL membuat Execution Plan:
- 1. Planner → memilih rencana tercepat
- 2. Executor → menjalankan rencana
- Analogi: mencari barang di supermarket - Sequential Scan (scan berurutan) vs Index Scan (katalog)

# Mengapa Query Bisa Lambat?

- Tabel terlalu besar → waktu lama untuk scan
- Tidak ada index → harus baca semua baris
- Subquery/join kompleks → banyak subquery (query di dalam query) atau banyak join antar tabel, maka PostgreSQL harus melakukan banyak langkah perhitungan di belakang layar.
- Statistik tabel tidak update → Sebuah tabel awalnya berisi 10.000 baris, Setelah beberapa bulan, tabelnya membengkak jadi 5 juta baris, Tapi statistiknya belum diperbarui, PostgreSQL masih mengira tabel itu kecil.

# Cara Membuat Query Lebih Cepat

- Gunakan Indexing
- JOIN efisien
- Hindari subquery berat
- Gunakan EXPLAIN / EXPLAIN ANALYZE

# Indexing

- Seperti daftar isi buku → langsung ke lokasi data
- `CREATE INDEX idx_jurusan ON mahasiswa(jurusan);`
- Index = shortcut pencarian cepat (tidak perlu membaca satu persatu halaman cukup daftar isi)
- Terlalu banyak index = beban saat INSERT/UPDATE



# JOIN Efisien

- Contoh: `SELECT m.nama, k.nama_kelas FROM mahasiswa m JOIN kelas k ON m.kelas_id = k.id;`
- Pastikan kolom JOIN terindeks
- PostgreSQL akan menggunakan Index Scan

# Hindari Subquery Berat

- Lambat: `SELECT nama FROM mahasiswa WHERE id IN (SELECT mahasiswa_id FROM nilai WHERE nilai >= 80);`
- Lebih cepat: `SELECT DISTINCT m.nama FROM mahasiswa m JOIN nilai n ON m.id = n.mahasiswa_id WHERE n.nilai >= 80;`
- `DISTINCT` digunakan untuk menghilangkan data duplikat (baris yang sama persis) dari hasil query.

# Gunakan EXPLAIN / EXPLAIN ANALYZE

- Lihat bagaimana query dijalankan:
- `EXPLAIN ANALYZE SELECT * FROM mahasiswa WHERE jurusan = 'Informatika';`
- Index Scan → cepat
- Seq Scan → baca semua baris (lambat)

# Primary Key vs Index

- Primary Key: unik, tidak boleh NULL, otomatis terindeks
- Index: bisa di kolom mana pun, mempercepat pencarian

# Tipe Index di PostgreSQL

- B-Tree → default, cocok untuk =, <, >
- Hash → exact match (=)
- GiST → data spasial, teks, range
- SP-GiST → data tersebar
- BRIN → tabel besar, data berurutan

# Ringkasan Fundamental

- Query → perintah ambil data
- Execution Plan → rencana eksekusi database
- Index → shortcut pencarian cepat
- JOIN/Subquery → kombinasi/filter data
- EXPLAIN/ANALYZE → analisis performa query



# Tips Tambahan

- ANALYZE nama\_tabel;
- Hindari SELECT \*
- Gunakan LIMIT saat debugging
- Pantau performa dengan pg\_stat\_statements

```
50 -- Penggunaan EXPLAIN / EXPLAIN ANALYZE
51 EXPLAIN
52 SELECT m.nama, m.jurusan, u.nama_dekan
53 FROM mahasiswa m
54 INNER JOIN universitas u
55 ON m.jurusan = u.jurusan; 1ms
56
57
```

Result(RO) X

Search Results

QUERY PLAN

- > Hash Join (cost=15.40..46.11 rows=456 width=294)
  - > Hash Cond: ((u.jurusan)::text = (m.jurusan)::text)
    - > -> Seq Scan on universitas u (cost=0.00..13.80 rows=380 width=176)
      - > -> Hash (cost=12.40..12.40 rows=240 width=176)
        - > -> Seq Scan on mahasiswa m (cost=0.00..12.40 rows=240 width=176)

1. PostgreSQL memilih **algoritma Hash Join** untuk menggabungkan dua tabel.
2. PostgreSQL membaca **seluruh isi tabel mahasiswa** (*Sequential Scan*).
3. Karena tidak ada index di kolom **jurusan**, PostgreSQL harus **membaca semua baris satu per satu**.
4. PostgreSQL membuat **struktur hash** dari hasil tabel **universitas**.
5. PostgreSQL membaca **seluruh isi tabel universitas** dari awal sampai akhir.

```
58 EXPLAIN ANALYZE
59 SELECT m.nama, m.jurusan, u.nama_dekan
60 FROM mahasiswa m
61 INNER JOIN universitas u
62 ON m.jurusan = u.jurusan; 9ms
```

Result(RO) X

Search Results

QUERY PLAN

- > Hash Join (cost=15.40..46.11 rows=456 width=294) (actual time=0.049..0.052 rows=7.00 loops=1)
  - > Hash Cond: ((u.jurusan)::text = (m.jurusan)::text)
  - > Buffers: shared hit=2
  - > -> Seq Scan on universitas u (cost=0.00..13.80 rows=380 width=176) (actual time=0.014..0.015 rows=6.00 loops=1)
    - > Buffers: shared hit=1
  - > -> Hash (cost=12.40..12.40 rows=240 width=176) (actual time=0.018..0.018 rows=10.00 loops=1)
    - > Buckets: 1024 Batches: 1 Memory Usage: 9kB
    - > Buffers: shared hit=1
  - > -> Seq Scan on mahasiswa m (cost=0.00..12.40 rows=240 width=176) (actual time=0.008..0.009 rows=10.00 loops=1)
    - > Buffers: shared hit=1
- > Planning:
  - > Buffers: shared hit=152
  - > Planning Time: 0.388 ms
  - > Execution Time: 0.090 ms

1. PostgreSQL melakukan join menggunakan **Hash Join**.
2. Setiap tabel dibaca penuh (karena **Sequential Scan**).
3. Query selesai dalam waktu sekitar **0.090 ms** (cepat karena datanya masih kecil).

Menambahkan **index** di kolom **jurusan** agar PostgreSQL bisa pakai **Index Scan**:

```
CREATE INDEX idx_jurusan_mhs ON mahasiswa(jurusan);  
CREATE INDEX idx_jurusan_univ ON universitas(jurusan);
```

## Fungsi Index Secara Umum

Sederhananya, **index itu seperti daftar isi di buku**.

### Tanpa index:

- Kalau kamu mencari kata “jurusan = 'Kimia'”, PostgreSQL harus **membuka semua halaman satu per satu** (scan seluruh tabel → **Seq Scan**).

### Dengan index:

- PostgreSQL langsung **melihat daftar isi (index)** untuk tahu posisi baris yang punya **jurusan = 'Kimia'**.
- Setelah itu, langsung **melompat ke lokasi baris tersebut** tanpa membaca seluruh tabel.

Hasilnya: **pencarian dan join jauh lebih cepat** apalagi kalau tabelnya besar.

```
60 EXPLAIN ANALYZE
61 SELECT m.nama, m.jurusan, u.nama_dekan
62 FROM mahasiswa m
63 INNER JOIN universitas u
64 ON m.jurusan = u.jurusan; 1ms
65
```

Result(RO) X

Search Results          Export   Cost: 1ms < 1 > Total 14

QUERY PLAN

- > Hash Join (cost=1.14..2.33 rows=6 width=294) (actual time=0.020..0.023 rows=7.00 loops=1)
  - > Hash Cond: ((m.jurusan)::text = (u.jurusan)::text)
  - > Buffers: shared hit=2
  - > -> Seq Scan on mahasiswa m (cost=0.00..1.10 rows=10 width=176) (actual time=0.006..0.007 rows=10.00 loops=1)
    - > Buffers: shared hit=1
  - > -> Hash (cost=1.06..1.06 rows=6 width=176) (actual time=0.009..0.009 rows=6.00 loops=1)
    - > Buckets: 1024 Batches: 1 Memory Usage: 9kB
    - > Buffers: shared hit=1
  - > -> Seq Scan on universitas u (cost=0.00..1.06 rows=6 width=176) (actual time=0.005..0.006 rows=6.00 loops=1)
    - > Buffers: shared hit=1
- > Planning:
  - > Buffers: shared hit=36 read=2
  - > Planning Time: 0.434 ms
  - > Execution Time: 0.046 ms

Query selesai dalam waktu sekitar **0.046 ms** lebih cepat dibandingkan tanpa menggunakan index sekitar **0.090 ms**

# THANKS!

arif.wicaksono@lecturer.itk.ac.id  
+62 852 1308 1309

