

JOIN QUERY DALAM SQL

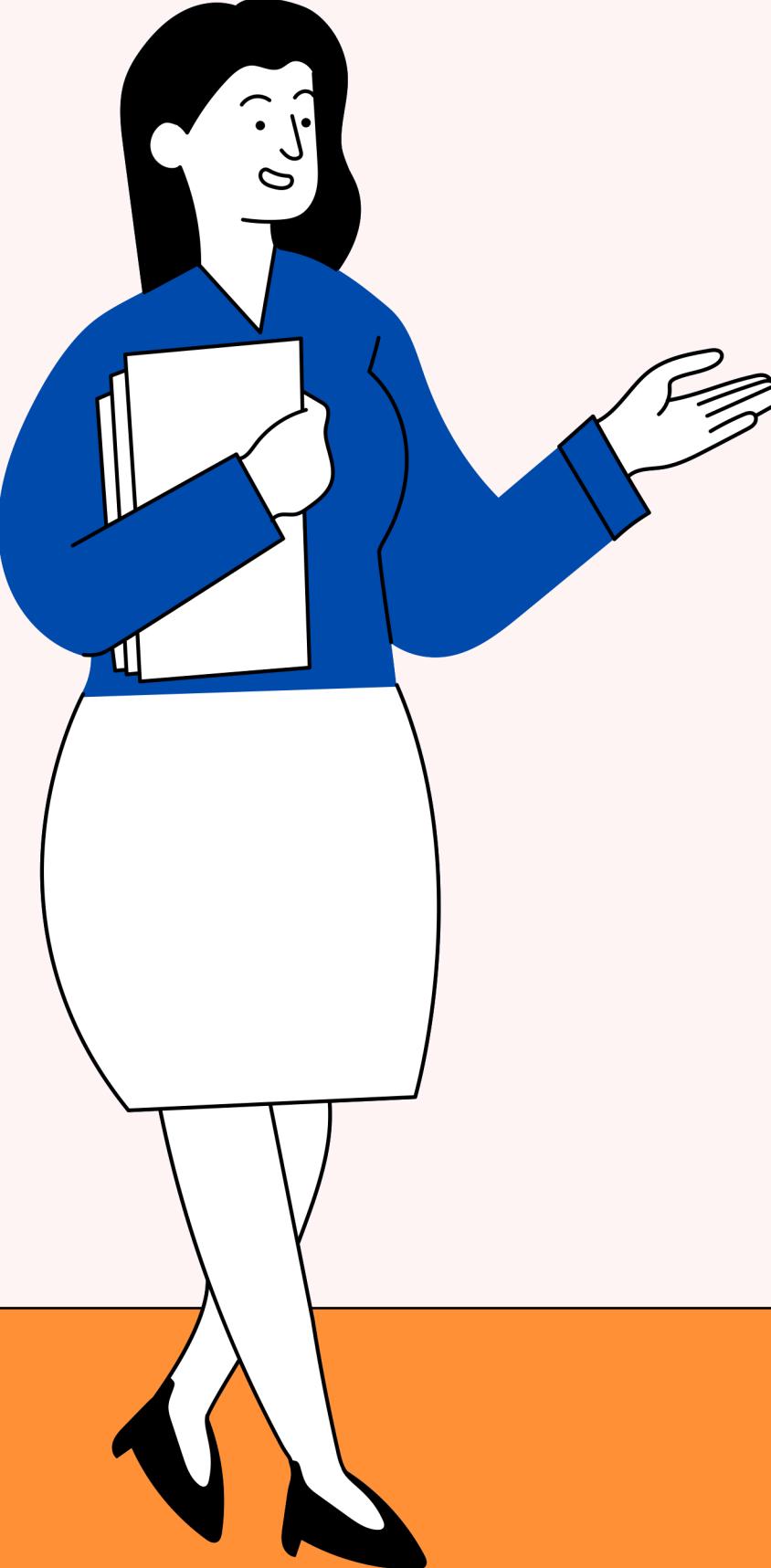
By Joni Syofian





Hello!

Dokumen ini merupakan catatan Saya dalam mempelajari join di sql. Semoga dapat bermanfaat dan kalau terdapat hal yang keliru mohon diluruskan.
Terima kasih 😊



Perintah JOIN dalam SQL digunakan untuk menampilkan data pada tabel yang saling berhubungan atau berelasi ataupun dengan tabel itu sendiri. Artinya kita dapat menampilkan data dalam beberapa tabel hanya dengan satu kali perintah. Berikut ini merupakan tipe-tipe join dalam SQL:

- **INNER JOIN atau JOIN**
- **LEFT JOIN atau LEFT OUTER JOIN**
- **RIGHT JOIN atau RIGHT OUTER JOIN**
- **FULL JOIN atau FULL OUTER JOIN**
- **CROSS JOIN**
- **SELF JOIN**
- **NATURAL JOIN**

Mari kita telusuri satu persatu!

Pendahuluan

INNER JOIN atau JOIN

Secara sederhananya adalah menggabungkan dua (atau lebih) tabel dengan key tertentu yang dimiliki oleh tabel yang akan dilakukan proses join.

Contoh ilustrasi dari tipe join ini adalah sebagai berikut:

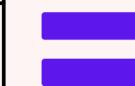
Tabel nama

nama_id	nama
1	Ratna
2	Ratih
3	Reza
4	Galih



Tabel pekerjaan

nama_id	Pekerjaan
1	Mahasiswa
3	Guru
4	Manager
5	Psikolog



Tabel hasil INNER JOIN

nama	Pekerjaan
Ratna	Mahasiswa
Reza	Guru
Galih	Manager

QUERY

```
SELECT n.nama,  
       p.pekerjaan  
FROM nama n  
JOIN pekerjaan p  
ON n.nama_id = p.nama_id;
```

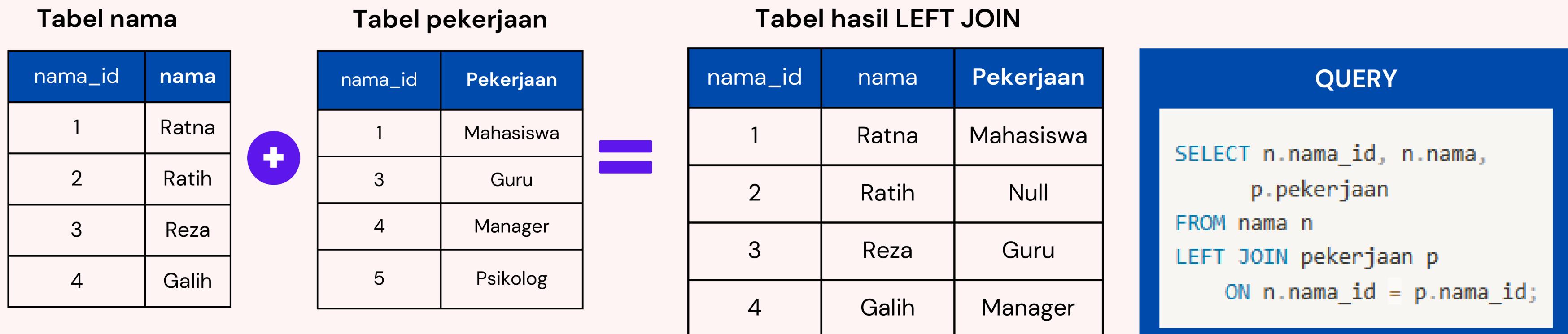
Dari contoh tabel di atas dapat dilihat key dari kedua tabel tersebut adalah **nama_id**. pada tabel nama terdapat **nama_id : 1, 2, 3, 4**. Sedangkan pada tabel pekerjaan terdapat **nama_id : 1, 3, 4, 5**. Sehingga tabel hasil join dari kedua tabel tersebut adalah yang memiliki **nama_id yang sama** yaitu **1, 3, dan 4**.

*pada tabel tertentu mungkin penamaan key diantara tabel berbeda. namun memiliki makna yang sama. Misal nama_id pada tabel nama diganti menjadi id. Sedangkan pada tabel pekerjaan tetap nama_id. Keduanya sebenarnya memiliki makna yang sama sehingga dapat dilakukan join berdasarkan key tersebut (id dan nama_id)



LEFT JOIN atau LEFT OUTER JOIN

Secara sederhananya adalah menggabungkan dua (atau lebih) dan mengambil semua nilai tabel sebelah kiri berdasarkan sebuah key dan tabel sebelah kanan akan bernilai NULL ketika tidak ada kesamaan key. Untuk lebih memahami dapat dilihat ilustrasi berikut:



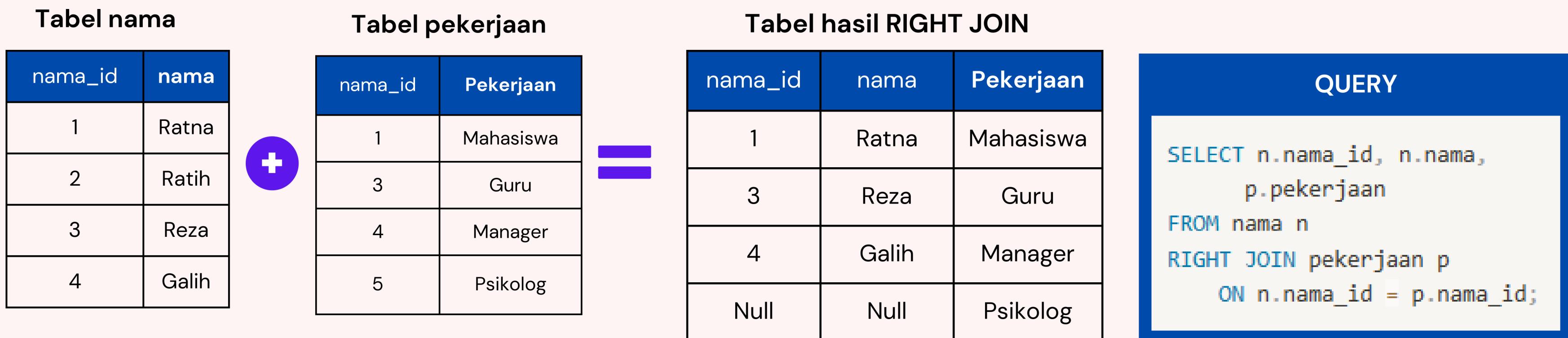
Dari contoh tabel di atas dapat dilihat key dari kedua tabel tersebut adalah **nama_id**. pada tabel nama (kiri) terdapat **nama_id : 1, 2, 3, 4**. Sedangkan pada tabel pekerjaan (kanan) terdapat **nama_id : 1, 3, 4, 5**. Sehingga tabel hasil join dari kedua tabel tersebut adalah yang memiliki **nama_id 1, 2, 3, 4** berdasarkan tabel nama. Pada **nama_id 2** pada kolom pekerjaan **bernilai NULL** karena **tidak terdapat data pekerjaan pada tabel pekerjaan dengan nama_id 2**.

*pada tabel tertentu mungkin penamaan key diantara tabel berbeda. namun memiliki makna yang sama. Misal nama_id pada tabel nama diganti menjadi id. Sedangkan pada tabel pekerjaan tetap nama_id. Keduanya sebenarnya memiliki makna yang sama sehingga dapat dilakukan join berdasarkan key tersebut (id dan nama_id)



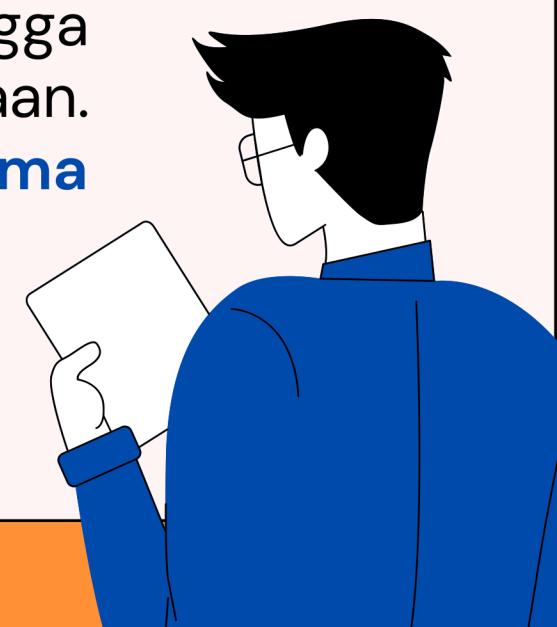
RIGHT JOIN atau RIGHT OUTER JOIN

RIGHT JOIN merupakan kebalikan dari LEFT JOIN. Dimana nilai yang akan diambil berdasarkan tabel sebelah kanan dan tabel sebelah kiri akan bernilai NULL ketika tidak ada kesamaan key dengan tabel sebelah kanan. Untuk lebih memahami dapat dilihat ilustrasi berikut:



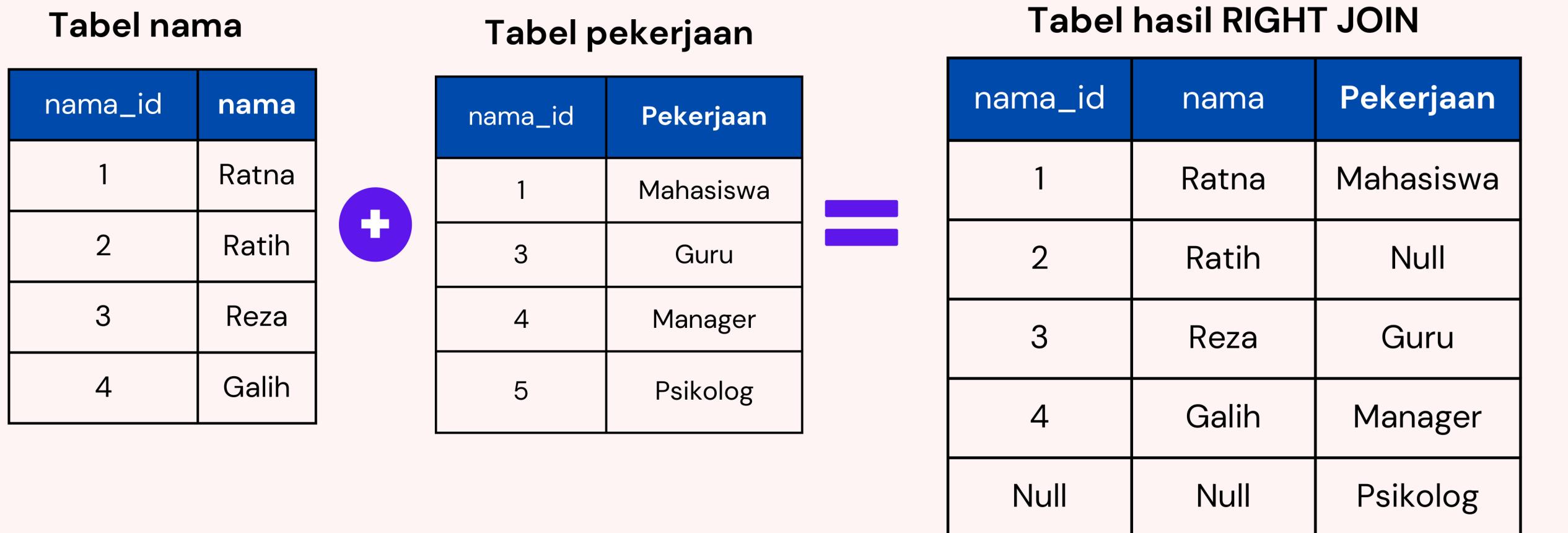
Dari contoh tabel di atas dapat dilihat key dari kedua tabel tersebut adalah **nama_id**. pada tabel nama (kiri) terdapat **nama_id : 1, 2, 3, 4**. Sedangkan pada tabel pekerjaan (kanan) terdapat **nama_id : 1, 3, 4, 5**. Sehingga tabel hasil join dari kedua tabel tersebut adalah yang memiliki **nama_id 1, 3, 4, 5** berdasarkan tabel pekerjaan. Pada **nama_id 5** yaitu pada kolom nama **bernilai NULL** hal ini disebabkan karena **tidak terdapat data nama pada tabel nama dengan nama_id 5**.

*pada tabel tertentu mungkin penamaan key diantara tabel berbeda. namun memiliki makna yang sama. Misal nama_id pada tabel nama diganti menjadi id. Sedangkan pada tabel pekerjaan tetap nama_id. Keduanya sebenarnya memiliki makna yang sama sehingga dapat dilakukan join berdasarkan key tersebut (id dan nama_id)



FULL JOIN atau FULL OUTER JOIN

Dengan FULL JOIN kita dapat memperoleh semua data dari tabel-tabel yang dilakukan join. Sehingga pada data dengan key yang tidak cocok antar tabelnya akan bernilai Null. Karena pada MySQL tidak dapat menggunakan FULL JOIN statement maka dapat dilakukan dengan UNION dari LEFT dan RIGHT JOIN seperti ilustrasi berikut:



Query

```
SELECT *  
FROM nama n  
LEFT JOIN pekerjaan p  
ON n.nama_id = p.nama_id;  
UNION ALL  
SELECT *  
FROM nama n  
RIGHT JOIN pekerjaan p  
ON n.nama_id = p.nama_id;  
WHERE n.nama_id IS NULL
```

Dari contoh tabel di atas dapat dilihat key dari kedua tabel tersebut adalah **nama_id**. pada tabel nama (kiri) terdapat **nama_id : 1, 2, 3, 4**. Sedangkan pada tabel pekerjaan (kanan) terdapat **nama_id : 1, 3, 4, 5**. Sehingga tabel hasil join dari kedua tabel tersebut adalah yang memiliki **nama_id 1, 2, 3, 4, 5 (kedua tabel)**, Dari sini dapat dilihat bahwa ketika tidak ada key yang cocok antar tabel, maka akan bernilai Null. Contoh di sini adalah **nama_id 2** yang **ada pada tabel nama** tetapi **tidak ada pada tabel pekerjaan** dan **nama_id 5** yang **ada pada tabel pekerjaan**, tetapi **tidak ada pada tabel nama**.

*pada tabel tertentu mungkin penamaan key diantara tabel berbeda. namun memiliki makna yang sama. Misal nama_id pada tabel nama diganti menjadi id. Sedangkan pada tabel pekerjaan tetap nama_id. Keduanya sebenarnya memiliki makna yang sama sehingga dapat dilakukan join berdasarkan key tersebut (id dan nama_id)



CROSS JOIN

Berbeda dengan tipe join yang lainnya, tipe join ini tidak membutuhkan suatu key. Hasil dari join ini adalah kombinasi dari kolom yang dilakukan cross join. Untuk lebih jelasnya dapat dilihat pada ilustrasi tabel berikut:

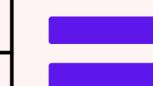
Tabel makanan

Makanan
Rendang
Sate



Tabel minuman

Minuman
Es Jeruk
Kopi
Teh



Tabel hasil CROSS JOIN

Makanan	Minuman
Rendang	Es Jeruk
Sate	Es Jeruk
Rendang	Kopi
Sate	Kopi
Rendang	Teh
Sate	Teh

QUERY

```
SELECT *
FROM makanan
CROSS JOIN minuman
```

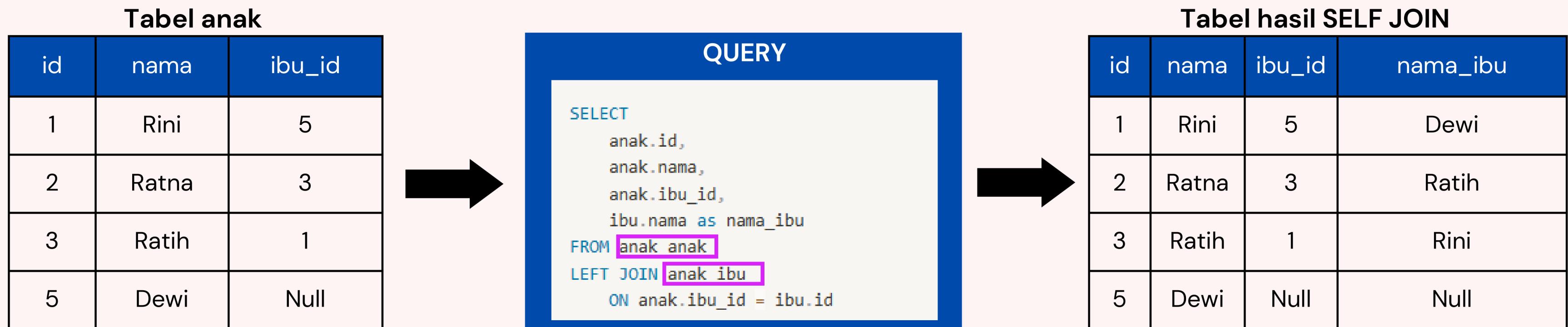
Dari ilustrasi di CROSS JOIN tersebut dapat dilihat bahwa **tabel yang dihasilkan** merupakan **kombinasi dari nilai-nilai yang ada pada kedua tabel** dan **tidak membutuhkan suatu key** yang menyatakan relasi antar dua tabel tersebut.



SELF JOIN

Self join, seperti namanya, **menggabungkan tabel dengan dirinya sendiri**. Untuk menggunakan gabungan mandiri, tabel harus berisi kolom (misal X) yang bertindak sebagai kunci utama dan kolom berbeda (misal Y) yang menyimpan nilai yang dapat dicocokkan dengan nilai di Kolom X. Nilai dari Kolom X dan Y tidak harus sama untuk setiap baris tertentu, dan nilai di Kolom Y dapat bernilai Null. Untuk lebih lanjutnya dapat dilihat pada ilustrasi permasalahan berikut.

Lakukan query untuk menentukan ibu dari nama-nama yang ada pada Tabel anak berikut:



Dari Ilustrasi dan query di atas dapat dilihat bahwa proses join dilakukan dengan tabel itu sendiri dalam hal ini berdasarkan **ibu_id**. Di sini digunakan **LEFT JOIN** sehingga ketika tidak ada informasi id maka akan tetap **menampilkan semua data**. Akan tetapi ketika dilakukan dengan **INNER JOIN** maka **id 5** tersebut **tidak akan masuk ke dalam hasil query**.



NATURAL JOIN

NATURAL JOIN adalah gabungan yang membuat gabungan implisit yang didasarkan pada kolom yang sama di tabel gabungan (penamaannya). selanjutnya, klausa gabungan digunakan untuk menggabungkan tabel berdasarkan kolom umum dan kondisi gabungan. Pada Natural JOIN tidak digunakan ON sebagai klausa antar tabelnya. Untuk lebih jelasnya dapat dilihat pada ilustrasi berikut:

item_id	nama_item	toko_id
1	Pensil	1
2	Buku	1
3	Tas	2
5	Sepatu	3



toko_id	nama_toko
1	Jaya
2	Makmur
3	Subur
4	Sejahtera



toko_id	item_id	nama_item	nama_toko
1	2	Buku	Jaya
1	1	Pensil	Jaya
2	3	Tas	Makmur
3	5	Sepatu	Subur

QUERY

```
SELECT *
FROM item
NATURAL JOIN toko;
```

Dari ilustrasi di atas dapat dilihat bahwa dalam proses NATURAL JOIN tidak diperlukan **klausa ON** melainkan **JOIN dapat dilakukan karena adanya kesamaan kolom antar dua tabel.**



Semoga
Bermanfaat!



Terima Kasih

Email
jonisyofian14@gmail.com

LinkedIn
<https://www.linkedin.com/in/jonisyofian/>