Operasi JOIN

Yance Sonatha, MT

Apa itu Join??

- Di dalam database, ada kalanya kita membutuhkan data dari beberapa tabel yang saling berhubungan. Untuk mendapatkan data dari beberapa tabel tersebut dapat digunakan perintah join pada perintah SQL.

- Join adalah cara untuk menghubungkan data yang diambil dari tabel-tabel melalui sebuah kolom yang menghubungkan mereka.

Mengapa join itu penting?

- **Join** memperbolehkan kita untuk mengambil data dari beberapa tabel melalui satu *query*. Hanya menggunakan sebuah tabel artinya kita hanya dapat menyimpan/memperoleh data yang terbatas atau justru menyimpan/memperoleh data yang terlalu banyak sehingga tabelnya menjadi kurang baik.
- **Join** menghubungkan satu tabel dengan tabel yang lain (inilah yang dimaksud dengan *relational* dari istilah *relational database*).

Jenis-Jenis Join

- Inner Join
- Outer Join
  - Full Outer Join
  - Right Outer Join
  - Left Outer Join
- Cross Join
- Union Join
- Self Join

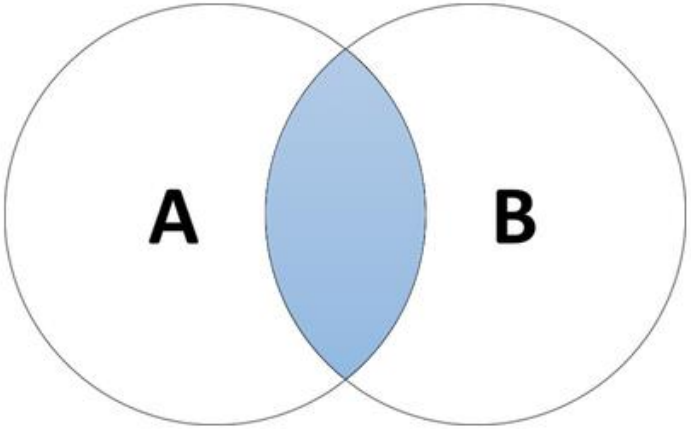
Inner Join

- **Inner join** merupakan jenis join yang paling umum yang dapat digunakan pada semua database. Jenis ini dapat digunakan bila ingin merelasikan dua set data yang ada di tabel, letak relasinya setelah pada perintah **ON pada join.**

- **Inner join** mengembalikan baris-baris dari dua tabel atau lebih yang memenuhi syarat.

Perintah Inner Join

```
SELECT field1,field2, fieldn
FROM tabel1 INNER JOIN tabel2
ON tabel1.key = tabel2.key
```



Outer Join

- Outer join merupakan jenis join yang sedikit berbeda dengan inner join. Pada MySQL, bentuk perintah untuk menerapkan outer join ada 2 yaitu :

1. Left [Outer] Join
2. Right [Outer]Join
3. Full [Outer] Join

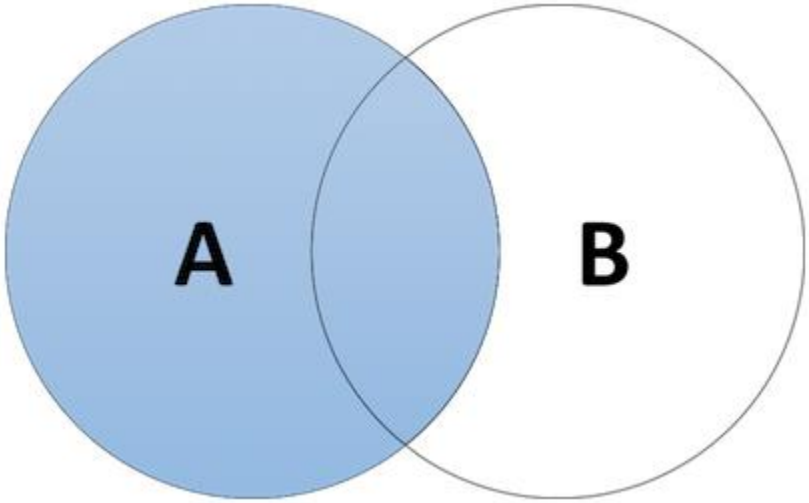
Left [Outer] Join

- **Left outer join** (sering disingkat **left join**) akan mengembalikan seluruh baris dari tabel disebelah *kiri* yang dikenai kondisi **ON** dan hanya baris dari tabel disebelah *kanan* yang memenuhi kondisi join.

- **Left join** Adalah Relasi Antar Table, biasanya Digunakan untuk menghasilkan baris data dari tabel kiri (nama tabel pertama/ Tabel Utama) yang tidak ada pasangan/Tidak Berelasi datanya pada tabel kanan (nama tabel kedua).

Perintah Left Join

```
SELECT field1,field2, fieldn
FROM tabel1 LEFT JOIN tabel2
ON tabel1.key = tabel2.key
```

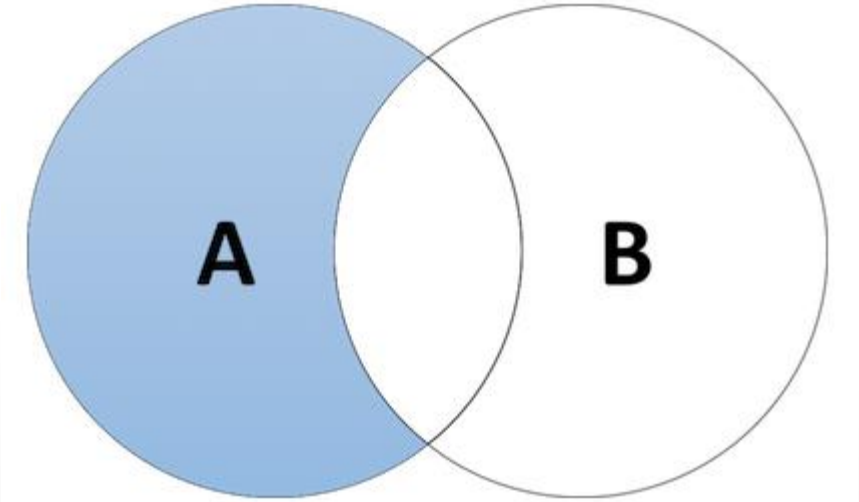


Left [Outer] Join without Intersection

- **Join** ini merupakan variasi dari *left outer join*. Pada **join** ini kita hanya akan mengambil data dari tabel sebelah kiri yang dikenai kondisi **ON** yang juga memenuhi kondisi *join* tanpa data dari tabel sebelah *kanan* yang memenuhi kondisi *join*.

Perintah Left Join Without Intersection

```
SELECT field1,field2, fieldn
FROM tabel1 LEFT JOIN tabel2
ON tabel1.key = tabel2.key
where tabel2.key is NULL
```



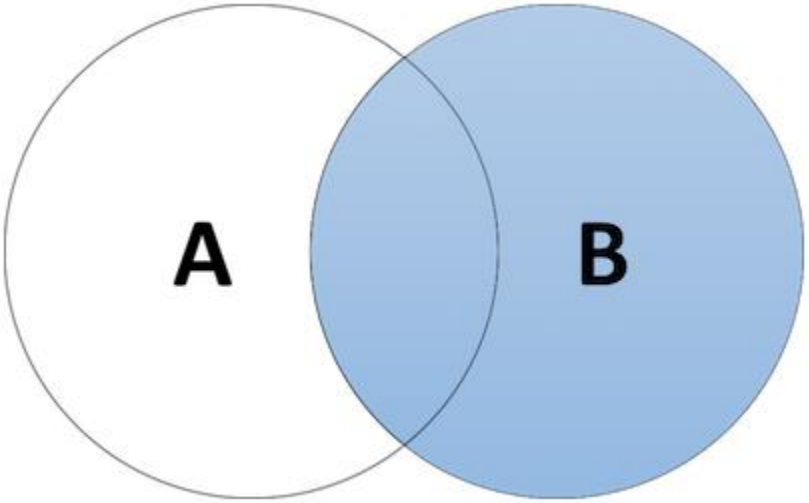
Right [Outer] Join

- **Right outer join** (sering disingkat **right join**) akan mengembalikan semua baris dari tabel sebelah kanan yang dikenai kondisi **ON** dengan data dari tabel sebelah kiri yang memenuhi kondisi *join*. Teknik ini merupakan kebalikan dari *left outer join*.

- **RIGHT JOIN** digunakan untuk menghasilkan baris data dari tabel kanan (nama tabel kedua/ Tabel Utama) yang tidak ada pasangan datanya/ Tidak Berelasi pada tabel kiri (nama tabel pertama).

Perintah Right Join

```
SELECT field1,field2, fieldn
FROM tabel1 RIGHT JOIN tabel2
ON tabel1.key = tabel2.key
```

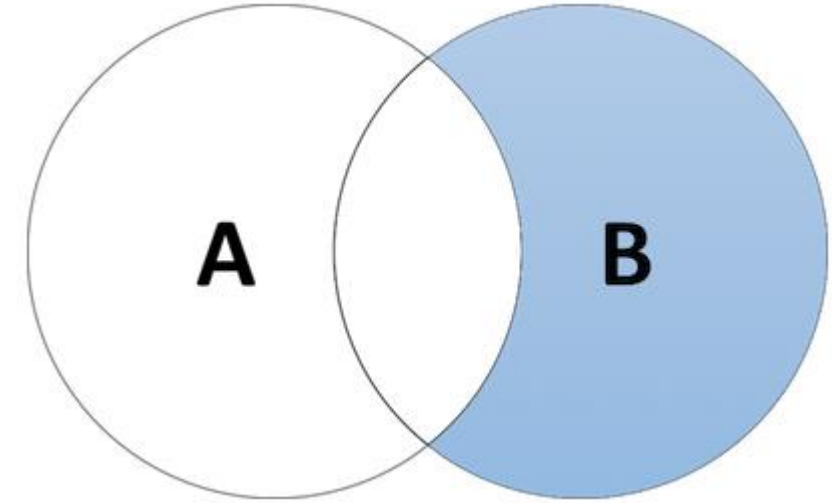


Right[Outer] Join without Intersection

- Teknik ini merupakan variasi dari *right outer join*. Pada **join** ini kita hanya akan mengambil data dari tabel sebelah kanan yang dikenai kondisi **ON** yang juga memenuhi kondisi *join* tanpa data dari tabel sebelah *kanan* yang memenuhi kondisi *join*.

Perintah Right Join Without Intersection

```
SELECT field1,field2, fieldn
FROM tabel1 RIGHT JOIN tabel2
ON tabel1.key = tabel2.key
where tabel1.key is NULL
```

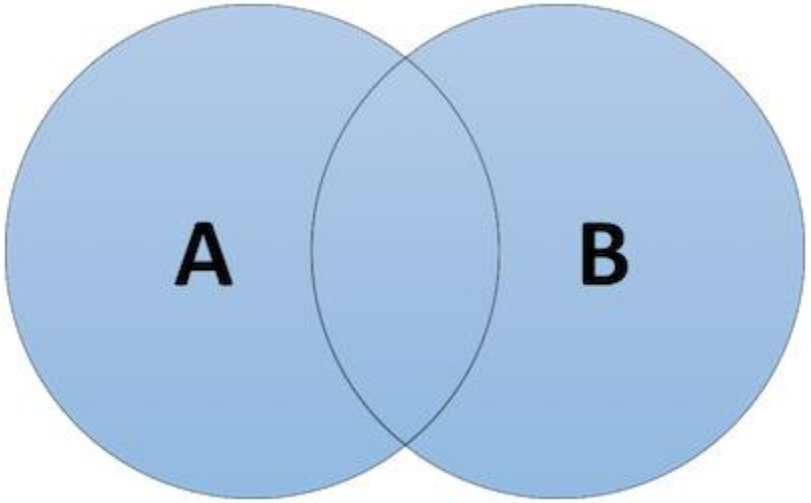


Full (Outer) Join

- **Full outer join** (sering disingkat **full join**) akan mengembalikan seluruh baris dari kedua tabel yang dikenai **ON** termasuk data-data yang bernilai NULL.

Perintah Full Join

```
SELECT field1,field2, fieldn
FROM tabel1 FULL OUTER JOIN tabel2
ON tabel1.key = tabel2.key
```

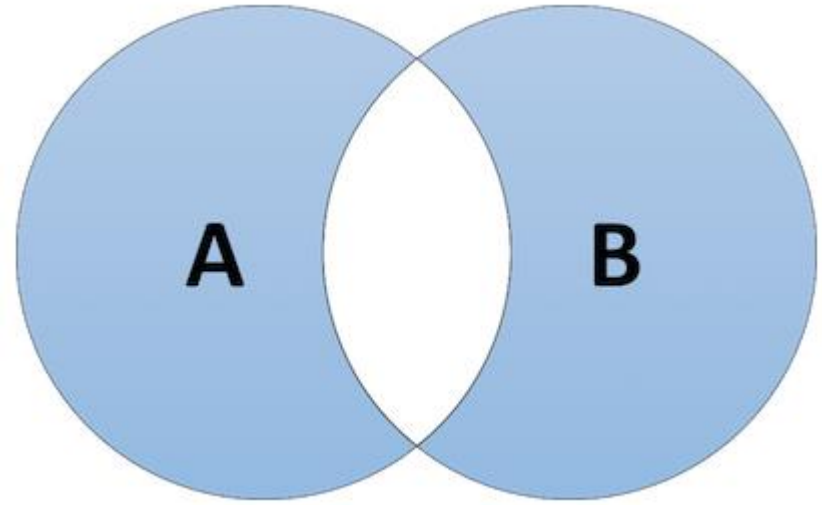


Full[Outer] Join without Intersection

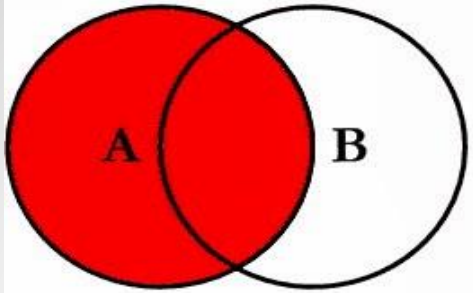
- Variasi lain dari *full outer join* yang akan mengembalikan seluruh data dari kedua tabel yang dikenai *ON* tanpa data yang memiliki nilai NULL.

Perintah Full Join Without Intersection

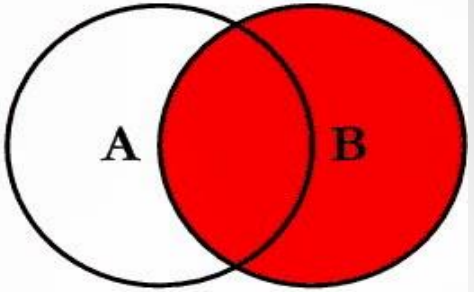
```
SELECT field1,field2, fieldn
FROM tabel1 FULL JOIN tabel2
ON tabel1.key = tabel2.key
where tabel1.key is NULL OR
tabel2.key is NULL
```



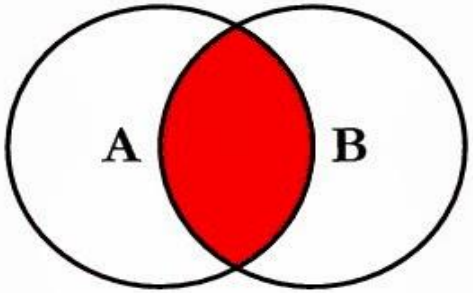
SQL JOINS



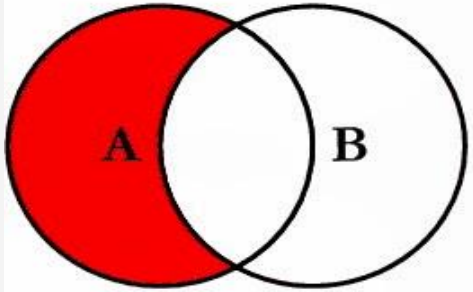

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



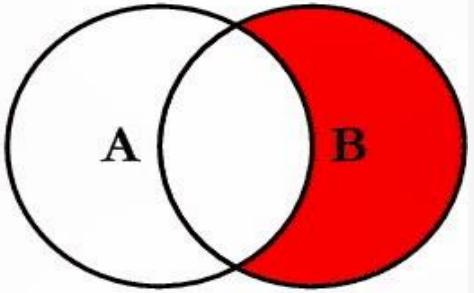

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



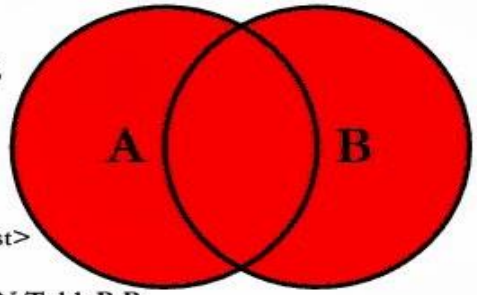

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



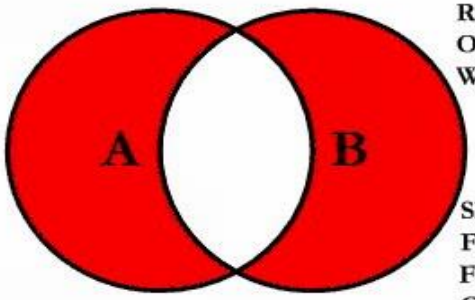

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
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
```

CROSS JOIN

- Cross join kadangkala disebut juga sebagai **Cartesian Product. Bila menggunakan cross join, maka** hasil dari cross join akan menciptakan hasil yang didasarkan pada semua kemungkinan kombinasi baris dalam kedua set data. Dengan demikian, jumlah baris yang dikembalikan adalah N M , dimana NX M adalah jumlah baris dalam kumpulan data A dan M jumlah baris dalam kumpulan data B. Jelas, jumlah baris dalam cross join dapat menjadi sampah.

Perintah Cross Join

SELECT field1,field2, fieldn
FROM tabel1 CROSS JOIN tabel2

atau

SELECT field1,field2, fieldn
FROM tabel1, tabel2

Union Join

- Union didukung oleh MySQL mulai dari versi 4.0. Pemakaian union dapat menyederhanakan perintah persyaratan **OR yang bertingkat. Bila dalam sebuah query menghasilkan pemakaian perintah OR yang** lebih dari satu sehingga dapat membuat bingung, sebagai gantinya digunakan perintah **UNION.**

- Union dapat dikatakan sebagai perintah untuk menggabungkan hasil query sql yang fungsinya sama dengan perintah OR.

Klasifikasi dan Perintah Union

- **Union Join**

SELECT *column_name(s)* FROM *table1*
UNION
SELECT *column_name(s)* FROM *table2*

- **Union All Join** : Berfungsi Untuk Menggabungkan Semua Data Pada Tabel Walaupun Isi Data Tabel Itu Sama.

SELECT *column_name(s)* FROM *table1*
UNION ALL
SELECT *column_name(s)* FROM *table2*

Self Join

- Perintah Join dimana suatu tabel berelasi dengan dirinya sendiri

SELECT ***column_name(s)***
FROM ***table1 T1, table1 T2***
WHERE ***condition***;