Rakamin Academy

investree

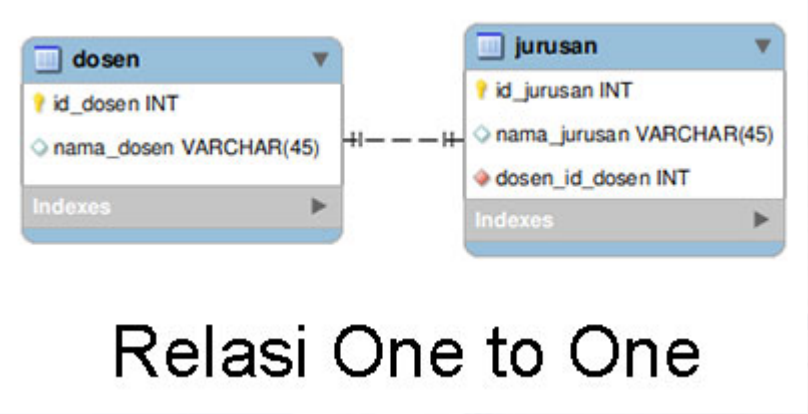
Virtual Internship Experience

MySQL: Table Relation and Join

1. Relasi antar table

a. Relasi *One to One*

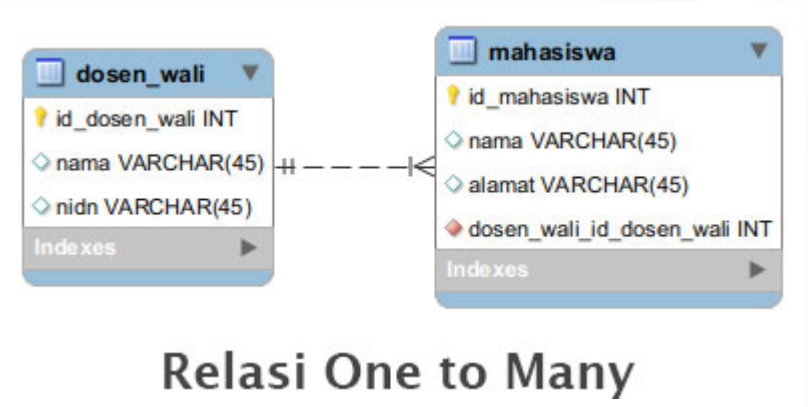
Relasi *One to One* adalah relasi yang mana setiap satu baris data pada tabel pertama hanya berhubungan dengan satu baris pada tabel kedua. Contohnya dibawah ini:



Pada tabel jurusan terdapat *primary key* **id_jurusan** dan *foreign key* **dosen_id_dosen**. Yang mana *foreign key* itulah yang digunakan sebagai penghubung tabel dosen.

b. Relasi *One to Many*

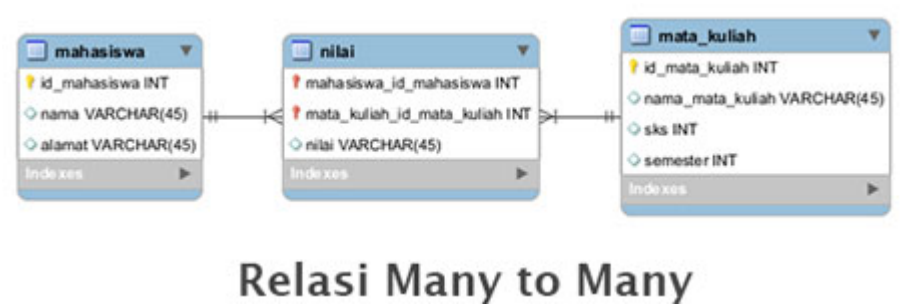
Relasi *One to Many* adalah relasi yang mana setiap satu baris data pada tabel pertama berhubungan dengan lebih dari satu baris pada tabel kedua. Contohnya di bawah ini:



Pada tabel mahasiswa terdapat *primary key* **id_mahasiswa** dan *foreign key* **dosen_wali_id_dosen_wali**. Yang mana *foreign key* itulah yang digunakan sebagai penghubung tabel dosen_wali

c. Relasi *Many to Many*

Relasi *Many to Many* adalah relasi yang mana setiap lebih dari satu baris data dari tabel pertama berhubungan dengan lebih dari satu baris data pada tabel kedua. Artinya, kedua tabel masing-masing dapat mengakses banyak data dari tabel yang direlasikan. Dalam hal ini, relasi *Many to Many* akan menghasilkan tabel ketiga sebagai perantara tabel kesatu dan tabel kedua sebagai tempat untuk menyimpan *foreign key* dari masing-masing tabel. Contohnya di bawah ini:



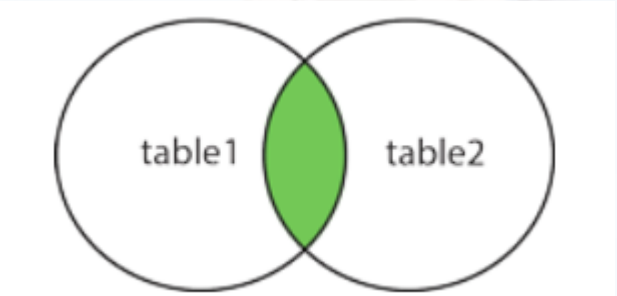
Relasi Many to Many

Coba perhatikan pada gambar di atas, terdapat tiga tabel yaitu tabel mahasiswa, nilai, dan mata_kuliah. Tabel mahasiswa dan mata kuliah tersebut masing-masing relasi *Many to Many* dan menghasilkan tabel baru yaitu tabel nilai. Sedangkan tabel baru atau tabel nilai tersebut sebagai penghubung antara tabel mahasiswa dan matakuliah yang mana tabel baru tersebut terdapat *foreign key* **mahasiswa_id_mahasiswa** dan **mata_kuliah_id_mata_kuliah** yang fungsinya untuk mengakses tabel mahasiswa dan matakuliah.

2. Cara query join

a. INNER JOIN

INNER JOIN membandingkan record di setiap table untuk dicek apakah nilai sama atau tidak.



Jika nilai kedua table sama maka akan terbentuk table baru yang hanya menampilkan record yang sama dari kedua table

Cara penulisannya:

```
SELECT *
FROM table1
INNER JOIN table2
ON table1.field = table2.field;
```

Contoh, Mencari data dari table mahasiswa dan tranksaksi berdasarkan kolom NIM

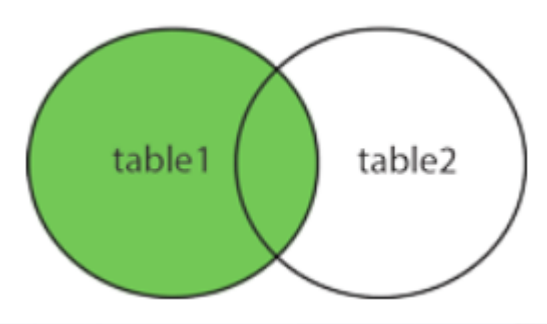
```
1  SELECT *
2  FROM mahasiswa
3  INNER JOIN transaksi
4  ON mahasiswa.nim = transaksi.nim;
5  +----------+-------+----------+--------------+----------+---------------
6  | nim      | nama  | alamat   | id_transaksi | nim      | buku
7  +----------+-------+----------+--------------+----------+---------------
8  | 21400200 | faqih | bandung  |            1 | 21400200 | Buku Informati
9  | 21400202 | anto  | semarang |            2 | 21400202 | Buku Teknik El
10 | 21400203 | dani  | padang   |            3 | 21400203 | Buku Matematik
11 | 21400206 | senta | semarang |            4 | 21400206 | Buku Fisika
12 +----------+-------+----------+--------------+----------+---------------
13 4 rows in set (0.00 sec)
```

Table mahasiswa mempunyai 7 record dan table transaksi mempunyai 7 record

Jika menggabungkan kedua data menggunakan INNER JOIN berdasarkan kolom NIM maka hanya tampil 4 data mahasiswa yang meminjam buku di perpustakaan

b. LEFT JOIN

LEFT JOIN menghasilkan nilai berdasarkan table kiri (table1) dan nilai yang sama di table kanan (table2).



Jika table kanan tidak nilainya ada maka akan diisi nilai NULL

```
SELECT *
FROM table1
LEFT JOIN table2
ON table1.field = table2.field;
```

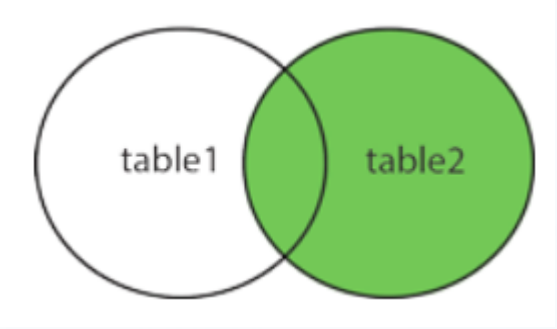
Contoh, Mencari data dari table mahasiswa dan tranksaksi berdasarkan kolom NIM

```
SELECT *
FROM mahasiswa
LEFT JOIN transaksi
ON mahasiswa.nim = transaksi.nim;
+----------+-------+----------+--------------+----------+---------------
| nim      | nama  | alamat   | id_transaksi | nim      | buku
+----------+-------+----------+--------------+----------+---------------
| 21400200 | faqih | bandung  |            1 | 21400200 | Buku Informati
| 21400202 | anto  | semarang |            2 | 21400202 | Buku Teknik El
| 21400203 | dani  | padang   |            3 | 21400203 | Buku Matematik
| 21400206 | senta | semarang |            4 | 21400206 | Buku Fisika
| 21400201 | ina   | jakarta  |         NULL |     NULL | NULL
| 21400204 | jaka  | bandung  |         NULL |     NULL | NULL
| 21400205 | nara  | bandung  |         NULL |     NULL | NULL
+----------+-------+----------+--------------+----------+---------------
7 rows in set (0.00 sec)
```

Table kiri (mahasiswa) akan menjadi table master dan mencari nilai yang sama di table transaksi. Apabila ada mahasiswa yang tidak meminjam buku maka diberi nilai NULL

c. RIGHT JOIN

Konsep RIGHT JOIN hampir sama seperti LEFT JOIN hanya yang menjadi master adalah table kanan (table 2)



Jika table kiri tidak nilainya ada maka akan diisi nilai NULL

```
SELECT *
FROM table1
RIGHT JOIN table2
ON table1.field = table2.field;
```

Contoh, Mencari data dari table mahasiswa dan tranksaksi berdasarkan kolom NIM

```
SELECT *
FROM mahasiswa
RIGHT JOIN transaksi
ON mahasiswa.nim = transaksi.nim;
+----------+-------+----------+-------------+----------+---------------
| nim      | nama  | alamat   | id_transaksi | nim      | buku
+----------+-------+----------+-------------+----------+---------------
| 21400200 | faqih | bandung  |           1 | 21400200 | Buku Informati
| 21400202 | anto  | semarang |           2 | 21400202 | Buku Teknik El
| 21400203 | dani  | padang   |           3 | 21400203 | Buku Matematik
| 21400206 | senta | semarang |           4 | 21400206 | Buku Fisika
|     NULL | NULL  | NULL     |           5 | 21400207 | Buku Bahasa In
|     NULL | NULL  | NULL     |           6 | 21400210 | Buku Bahasa Da
|     NULL | NULL  | NULL     |           7 | 21400211 | Buku Kimia
+----------+-------+----------+-------------+----------+---------------
7 rows in set (0.00 sec)
```

Terdapat 7 transaksi peminjaman buku di perpustakaan. Bagi transaksi yang NIM mahasiswa tidak ada di table mahasiswa akan diberi nilai NULL

3. Table view di mysql

VIEW adalah perintah untuk membuat table virtual yang menyimpan kode SQL. Dengan **view** kita bisa membuat kode SQL yang komplek dikemas menjadi satu table sederhana

```
CREATE VIEW <nama view>
AS
Kode SQL
```

Saat kita mengeksekusi CREATE VIEW maka akan terbentuk table virtual yang menyimpan kode SQL

Contoh, Kita akan membuat kode SQL yang menghubungkan tabel mahasiswa dan table transaksi secara INNER JOIN dan menyimpannya ke **view**

```
1  CREATE VIEW transaksiMhs AS
2  SELECT mahasiswa.nim, nama, alamat, buku
3  FROM mahasiswa
4  INNER JOIN transaksi
5  ON mahasiswa.nim = transaksi.nim
```

Daftar Pustaka

https://www.hostinger.co.id/tutorial/apa-itu-mysql
https://www.sekawanmedia.co.id/blog/pengertian-mysql/
https://www.angon.co.id/phpmysql/macam-macam-query-mysql-dan-fungsinya
https://aantamim.id/relasi-tabel-database/
https://ngodingdata.com/