Database

- A **database** is *an organized collection of data*.

- The **main purpose** of database is to operate large amount of information by storing, retrieving and managing.

- There are many dynamic websites on the world wide web nowadays which are handled through databases. For example, a model to checks the availability of rooms in a hotel. It is an example of dynamic website that uses database.

Purwadhika
Startup and Coding School

RDBMS

- **RDBMS** (*Relational Database Management Systems*) is database relational model based, introduced by E.F. Codd (1970s).

- In RDBMS, data is represented in terms of tuples (rows). It contains number of tables and each table has its own primary key.

- All modern database management systems like SQL, MS SQL Server, IBM DB2, ORACLE, My-SQL and Microsoft Access are based on RDBMS.

Purwadhika
Startup and Coding School

SQL

- **SQL** (*Structured Query Language*) is used to communicate with a database. It's the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database.

- Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc.

- The standard SQL commands such as "Select", "Insert", "Update", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database.

Purwadhika
Startup and Coding School

MySQL

MySQL is the world's most popular open source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, Yahoo and many more.

MySQL has been developed by MySQL AB & Oracle Corporation since 23rd May 1995, written in C and C++.

Purwadhika
Startup and Coding School

MySQL Ranking
2nd All DB-engines

341 systems in ranking, March 2018

	Rank		DBMS	Database Model	Score		
Mar 2018	Feb 2018	Mar 2017			Mar 2018	Feb 2018	Mar 2017
1.	1.	1.	Oracle ➕	Relational DBMS	1289.61	-13.67	-109.89
2.	2.	2.	MySQL ➕	Relational DBMS	1228.87	-23.60	-147.21
3.	3.	3.	Microsoft SQL Server ➕	Relational DBMS	1104.79	-17.25	-102.70
4.	4.	4.	PostgreSQL ➕	Relational DBMS	399.35	+10.97	+41.71
5.	5.	5.	MongoDB ➕	Document store	340.52	+4.10	+13.59
6.	6.	6.	DB2 ➕	Relational DBMS	186.66	-3.31	+1.75
7.	7.	7.	Microsoft Access	Relational DBMS	131.95	+1.88	-0.99
8.	8.	⬆ 10.	Redis ➕	Key-value store	131.22	+4.21	+18.22
9.	9.	⬆ 11.	Elasticsearch ➕	Search engine	128.54	+3.23	+22.32
10.	10.	⬇ 8.	Cassandra ➕	Wide column store	123.49	+0.71	-5.70

https://db-engines.com/en/ranking

Purwadhika
Startup and Coding School

MySQL Ranking
2nd RDBMS DB-engines

137 systems in ranking, March 2018

	Rank		DBMS	Database Model	Score		
Mar 2018	Feb 2018	Mar 2017			Mar 2018	Feb 2018	Mar 2017
1.	1.	1.	Oracle 🔶	Relational DBMS	1289.61	-13.67	-109.89
2.	2.	2.	MySQL 🔶	Relational DBMS	1228.87	-23.60	-147.21
3.	3.	3.	Microsoft SQL Server 🔶	Relational DBMS	1104.79	-17.25	-102.70
4.	4.	4.	PostgreSQL 🔶	Relational DBMS	399.35	+10.97	+41.71
5.	5.	5.	DB2 🔶	Relational DBMS	186.66	-3.31	+1.75
6.	6.	6.	Microsoft Access	Relational DBMS	131.95	+1.88	-0.99
7.	7.	7.	SQLite 🔶	Relational DBMS	114.81	-2.46	-1.37
8.	8.	8.	Teradata	Relational DBMS	72.46	-0.53	-1.07
9.	⬆ 10.	⬆ 12.	MariaDB 🔶	Relational DBMS	63.10	+1.45	+16.22
10.	⬇ 9.	⬇ 9.	SAP Adaptive Server 🔶	Relational DBMS	62.62	-0.87	-7.51

https://db-engines.com/en/ranking

Purwadhika
Startup and Coding School

Primary Key

- The ***Primary keys*** constraint uniquely identifies each record in a database table.

- Primary keys must contain UNIQUE values, and cannot contain NULL values.

- A table can have only one primary key, which may consist of single or multiple fields.

- Example: *No_KTP, product_key, ID_sidikjari*

Purwadhika
Startup and Coding School

Composite Key

■ A ***composite key*** is a combination of two or more columns in a table that can be used to uniquely identify each row in the table when the columns are combined uniqueness is guaranteed, but when it taken individually it does not guarantee uniqueness.

■ Columns that make up the composite key can be of different data types.

■ Example: ***nama & tglLahir, ID_prod & prod_date***

■ A ***foreign key*** is a field or a column that is used to establish a link between two tables. In simple words you can say that, a foreign key in one table used to point primary key in another table.

Purwadhika
Startup and Coding School

Foreign Key

First table:

S_Id	LastName	FirstName	CITY
1	MAURYA	AJEET	ALLAHABAD
2	JAISWAL	RATAN	GHAZIABAD
3	ARORA	SAUMYA	MODINAGAR

Second table:

O_Id	OrderNo	S_Id
1	99586465	2
2	78466588	2
3	22354846	3
4	57698656	1

The **"S_Id"** column in the 1st table is the **PRIMARY KEY** in the 1st table.
The **"S_Id"** column in the 2nd table is a **FOREIGN KEY** in the 2nd table.

Purwadhika
Startup and Coding School

Getting Started

Download & install MySQL
from its official site *or* XAMPP!

MySQL Installer
www.mysql.com/
downloads/

*On Windows, it needs
.NET Framework*
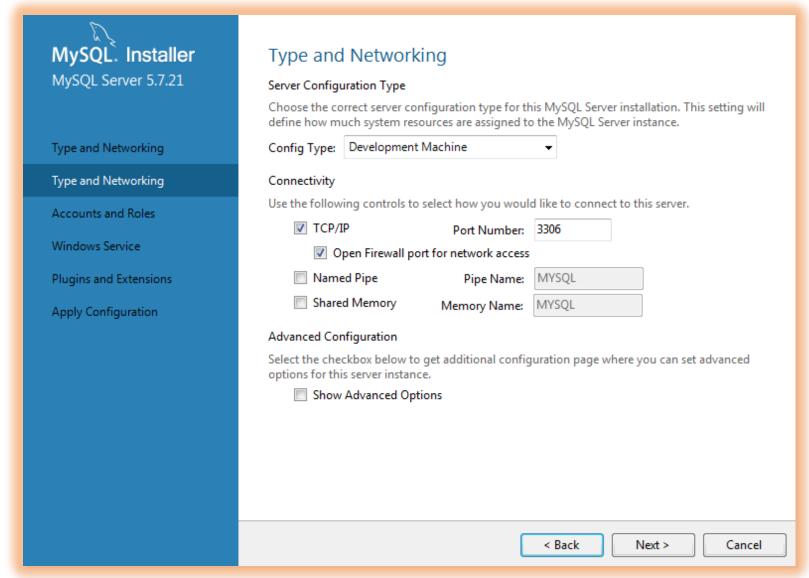www.microsoft.com/net/
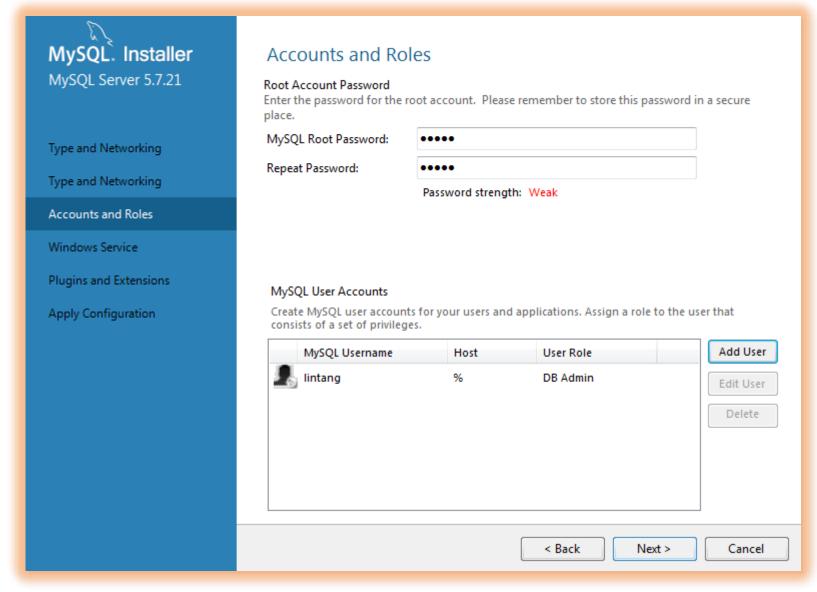download/windows

XAMPP Bundle
apachefriends.org

Purwadhika
Startup and Coding School

MySQL. Installer
MySQL Server 5.7.21

Type and Networking

Type and Networking

Accounts and Roles

Windows Service

Plugins and Extensions

Apply Configuration

Type and Networking

Server Configuration Type

Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.

Config Type: Development Machine ▾

Connectivity

Use the following controls to select how you would like to connect to this server.

☑ TCP/IP   Port Number: 3306

   ☑ Open Firewall port for network access

☐ Named Pipe   Pipe Name: MYSQL

☐ Shared Memory   Memory Name: MYSQL

Advanced Configuration

Select the checkbox below to get additional configuration page where you can set advanced options for this server instance.

☐ Show Advanced Options

[< Back]   [Next >]   [Cancel]

Via Command Prompt, go to *mysql.exe* directory
C:\Program Files\MySQL\MySQL Server 5.7\bin
then execute:

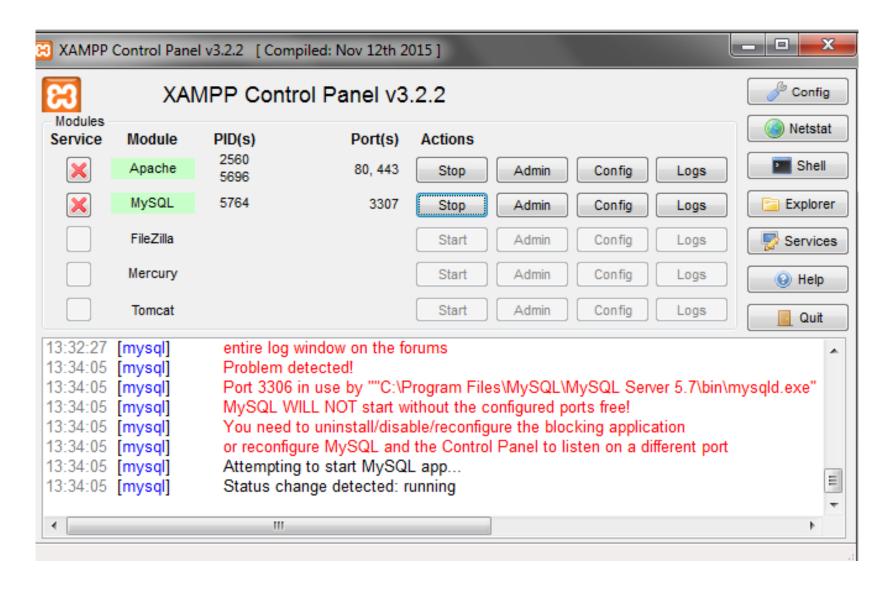**$ mysql.exe --user=root --password=12345**

or

$ mysql.exe -u lintang -p12345

```
C:\Program Files\MySQL\MySQL Server 5.7\bin>mysql.exe --user=lintang --password=12345
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 5.7.21-log MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> _
```

#2 With XAMPP
Activate Apache & MySQL Server

Via Command Prompt, go to *mysql.exe* directory
C:\xampp\mysql\bin then execute:

$ mysql.exe --user=root --password=12345

or

$ mysql.exe --user=lintang --password=12345

```
C:\xampp\mysql\bin>mysql.exe --user=root --password=12345
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 32
Server version: 5.7.21-log MySQL Community Server (GPL)

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

Create Database

Menampilkan daftar database yang ada:

```
$ SHOW databases;
```

Membuat database bernama "toko":

```
$ CREATE database toko;
$ CREATE database IF NOT EXISTS toko;
```

Menghapus database "toko":

```
$ DROP database toko;
```

Purwadhika
Startup and Coding School

Create Table

Menggunakan/mengaktifkan database "toko":

```
$ USE toko;
```

Menampilkan tabel dalam database "toko":

```
$ SHOW tables;
$ SHOW FULL tables;
```

Membuat tabel "tes" berisi 1 kolom "NoID":

```
$ CREATE TABLE tes (NoID INT);
```

Lihat struktur tabel "tes":

```
$ DESCRIBE tes;
```

Hapus tabel "tes":

```
$ DROP TABLE tes;
```

```
$ CREATE TABLE karyawan (
-> No INT NOT NULL AUTO_INCREMENT,
-> Nama VARCHAR(30) NOT NULL,
-> Usia TINYINT,
-> Berat FLOAT(3,1),
-> Kota TEXT,
-> Th YEAR,
-> Data_In TIMESTAMP,
-> PRIMARY KEY (No)
-> );
```

Purwadhika
Startup and Coding School

Insert Data

```
$ INSERT INTO karyawan VALUES(
-> NULL,
-> 'Andi Santosa',
-> 42,
-> 78.2,
-> 'Jakarta',
-> 2001,
-> NULL
-> );
```

Lihat semua data pada tabel karyawan:
```
$ SELECT * FROM karyawan;
```

Create Table

```
$ CREATE TABLE tim_sales (
-> No INT NOT NULL AUTO_INCREMENT,
-> Nama VARCHAR(30) NOT NULL,
-> Usia SMALLINT,
-> Sex SET('Pria','Wanita'),
-> Area ENUM('Jkt','Bdg','Sby'),
-> Kode VARCHAR(8) NOT NULL UNIQUE,
-> Gaji INT DEFAULT 3000000,
-> PRIMARY KEY (No)
-> );
```

//coba insert data

Purwadhika
Startup and Coding School

Insert Data

```
mysql> select * from karyawan;
+----+-----------------+------+-------+------------+------+---------------------+
| No | Nama            | Usia | Berat | Kota       | Th   | Data_In             |
+----+-----------------+------+-------+------------+------+---------------------+
|  1 | Andi Santosa    |   42 |  78.2 | Jakarta    | 2001 | 2018-02-17 21:36:33 |
|  2 | Budi Permana    |   32 |  88.1 | Jakarta    | 1999 | 2018-02-17 21:36:33 |
|  3 | Cecep Sutisna   |   35 |  68.9 | Bandung    | 2000 | 2018-02-17 21:36:33 |
|  4 | Dedi Hartanto   |   32 |  71.3 | Salatiga   | 1998 | 2018-02-17 21:36:33 |
|  5 | Eva Soraya      |   32 |  48.1 | Medan      | 2015 | 2018-02-17 21:36:33 |
|  6 | Farah Naimah    |   29 |  52.3 | Surabaya   | 2010 | 2018-02-17 21:36:33 |
|  7 | Gianti Safitri  |   22 |  51.6 | Bandung    | 2017 | 2018-02-17 21:36:33 |
|  8 | Hamzah Syah     |   34 |  66.0 | Yogyakarta | 2008 | 2018-02-17 21:36:33 |
|  9 | Irene Sukindar  |   25 |  49.3 | Jakarta    | 2016 | 2018-02-17 21:36:33 |
| 10 | Joni Saputra    |   28 |  69.8 | Yogyakarta | 2000 | 2018-02-17 21:36:33 |
+----+-----------------+------+-------+------------+------+---------------------+
10 rows in set (0.06 sec)
```

Purwadhika
Startup and Coding School

Insert Multiple Data

```
$ INSERT INTO karyawan VALUES
(NULL,'Andi Santosa',42,78.2,'Jakarta',2001,NULL),
(NULL,'Budi Permana',32,88.1,'Jakarta',1999,NULL),
(NULL,'Cecep Sutisna',35,68.9,'Bandung',2000,NULL),
(NULL,'Dedi Hartanto',32,71.3,'Salatiga',1998,NULL),
(NULL,'Eva Soraya',32,48.1,'Medan',2015,NULL),
(NULL,'Farah Naimah',29,52.3,'Surabaya',2010,NULL),
(NULL,'Gianti Safitri',22,51.6,'Bandung',2017,NULL),
(NULL,'Hamzah Syah',34,66.0,'Yogyakarta',2008,NULL),
(NULL,'Irene Sukindar',25,49.3,'Jakarta',2016,NULL),
(NULL,'Joni Saputra',28,69.8,'Yogyakarta',2000,NULL)
;
```

Insert Data

Pada tabel "karyawan" di database "toko",
Insert data hanya ke kolom Nama & Usia:

```
$ INSERT INTO karyawan (Nama, Usia) VALUES
-> ('Budi Raharja', 52);
```

Insert multiple data ke kolom Nama:

```
$ INSERT INTO karyawan (Nama) VALUES
-> ('Caca'),('Dedi');
```

Purwadhika
Startup and Coding School

Update Data

Update data untuk semua isi kolom tertentu:

```
$ UPDATE karyawan SET Th = 2010;
```

Update data untuk baris data tertentu:

```
$ UPDATE karyawan SET Nama = "Anisa Safitri"
-> WHERE No = 1;
```

```
$ UPDATE karyawan SET
  Nama = "Anisa Safitri", Usia = 28,
  Kota = "Medan", Job = "Programmer"
-> WHERE No = 1;
```

Delete Data

Delete semua data pada tabel:

```
$ DELETE FROM karyawan;
```

Delete baris data tertentu di tabel:

```
$ DELETE FROM karyawan
-> WHERE No = 4;
```

Insert Column

Lihat struktur tabel
```
$ DESCRIBE karyawan;
```

Add kolom Gaji:
```
$ ALTER TABLE karyawan
-> ADD COLUMN Gaji INT;
```

Add kolom Gaji setelah kolom Tahun:
```
$ ALTER TABLE karyawan
-> ADD COLUMN Gaji INT AFTER Th;
```

Hapus kolom:
```
$ ALTER TABLE karyawan
-> DROP COLUMN Gaji;
```

Select

Lihat semua isi data pada tabel:
```
$ SELECT * FROM karyawan;
```

Lihat data pada atribut/kolom tertentu:
```
$ SELECT Nama, Kota FROM karyawan;
```

Lihat 5 data pertama:
```
$ SELECT * FROM karyawan LIMIT 5;
```

Lihat 5 data setelah data ke-3:
```
$ SELECT * FROM karyawan LIMIT 3,5;
```

Purwadhika
Startup and Coding School

Select

Lihat additional data di luar tabel.

*Tampilkan setengah Berat Badan:

```
$ SELECT Nama, Berat,
-> 0.5 * Berat AS separoBB
-> FROM karyawan;
```

*Berapa tahun lagi waktu pensiun:

```
$ SELECT Nama, Usia,
-> 55 – Usia AS jarakPensiun
-> FROM karyawan;
```

Order By

*Urutkan Nama berdasarkan Usia:

```
$ SELECT Nama, Usia
-> FROM karyawan ORDER BY Usia;
```

*Urutan Descending Nama berdasar Usia:

```
$ SELECT Nama, Usia
-> FROM karyawan ORDER BY Usia DESC;
```

*Urutkan Nama berdasarkan Usia lalu Berat:

```
$ SELECT Nama, Usia, Berat
-> FROM karyawan ORDER BY Usia, Berat;
```

*Urutkan Nama & kelompokkan Nama
 berdasarkan urutan Kota:

```
$ SELECT Nama, Kota
-> FROM karyawan ORDER BY FIELD(Kota,
-> 'Jakarta','Yogyakarta','Bandung',
-> 'Salatiga','Medan','Surabaya');


$ SELECT Nama, Kota
-> FROM karyawan ORDER BY FIELD(Kota,
-> 'Jakarta','Yogyakarta','Bandung',
-> 'Salatiga','Medan','Surabaya')
-> DESC;
```

Where

```
$ SELECT * FROM karyawan
-> WHERE No > 2;

$ SELECT * FROM karyawan
-> WHERE Berat BETWEEN 45 AND 55;

$ SELECT Nama FROM karyawan
-> WHERE Usia < 25 OR Th > 2016;

$ SELECT Nama FROM karyawan
-> WHERE Th IN (2010,2017);
```

Where Like

```
$ SELECT Nama FROM karyawan
-> WHERE Nama LIKE 'a%';

$ SELECT Nama FROM karyawan
-> WHERE Nama LIKE '%na';

$ SELECT Nama FROM karyawan
-> WHERE Nama LIKE '%di%';

$ SELECT Nama FROM karyawan
-> WHERE Nama NOT LIKE '%di%';
```

Purwadhika
Startup and Coding School

Count, Min & Max

```
$ SELECT COUNT(*) FROM karyawan
-> WHERE Kota = 'Jakarta';

$ SELECT COUNT(*) AS Banyaknya
-> FROM karyawan
-> WHERE Kota = 'Jakarta';

$ SELECT MIN(Usia) AS Termuda
-> FROM karyawan;

$ SELECT MAX(Usia) AS Tertua
-> FROM karyawan;
```

Purwadhika
Startup and Coding School

Sum & Avg

```
$ SELECT SUM(Usia) FROM karyawan
-> WHERE Kota = 'Yogyakarta';


$ SELECT SUM(Usia) AS Total_Usia
-> FROM karyawan
-> WHERE Kota = 'Yogyakarta';


$ SELECT AVG(Usia) AS Rata_Usia
-> FROM karyawan
-> WHERE Kota = 'Yogyakarta';
```

Group By

Hitung banyaknya karyawan di tiap kota:

```
$ SELECT Kota, COUNT(*) AS Karyawan
-> FROM karyawan
-> GROUP BY Kota;
```

Hitung rerata BB karyawan di tiap kota:

```
$ SELECT Kota, AVG(Berat) AS Rata_BB
-> FROM karyawan
-> GROUP BY Kota;
```

Having

Rerata BB karyawan di tiap kota:

```
$ SELECT Kota, AVG(Berat) AS Rata_BB
-> FROM karyawan
-> GROUP BY Kota;
```

Tampilkan kota dg rerata BB kary < 65:

```
$ SELECT Kota, AVG(Berat) AS Rata_BB
-> FROM karyawan
-> GROUP BY Kota
-> HAVING Rata_BB < 65;
```

Subquery
Query inside query

Menampilkan nama & usia karyawan di Jakarta, yang usianya > rata-rata usia karyawan di Yogyakarta:

```
$ SELECT Nama, Usia FROM karyawan
-> WHERE Kota = 'Jakarta' AND Usia > (
-> SELECT AVG(Usia) FROM karyawan
-> WHERE Kota = 'Yogyakarta');
```

Purwadhika
Startup and Coding School

Subquery From

Menampilkan rata-rata usia dari:
tabel nama & usia karyawan di Jakarta,
yang usianya > rata-rata usia
karyawan di Yogyakarta:

```
$ SELECT AVG(Usia) AS Rata_Usia FROM
-> (SELECT Nama, Usia FROM karyawan
-> WHERE Kota = 'Jakarta' AND Usia > (
-> SELECT AVG(Usia) FROM karyawan
-> WHERE Kota = 'Yogyakarta'));
```

Purwadhika
Startup and Coding School

File   Edit   View   Database   Tools   Scripting   Help

Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

Browse Documentation >   Read the Blog >   Discuss on the Forums >

MySQL Connections ⊕ ⊘   ⚲ Filter connections

Local instance MySQL57
👤 root
localhost:3306

Purwadhika
Startup and Coding School

Working with GUI
#3 Happy Querying!

MySQL Workbench

Local instance MySQL57 (sys)   ×   Local instance MySQL57 (toko) ×

File   Edit   View   Query   Database   Server   Tools   Scripting   Help

Query 1 ×

Limit to 1000 rows

```
 1  CREATE TABLE karyawan (
 2    No INT NOT NULL AUTO_INCREMENT,
 3    Nama VARCHAR(30) NOT NULL,
 4    Usia SMALLINT,
 5    Berat FLOAT(3,1),
 6    Kota TEXT,
 7    Th YEAR,
 8    Data_In TIMESTAMP,
 9    PRIMARY KEY (No)
10  );
11  SHOW tables;
12  DESCRIBE karyawan;
```

Insert & execute query here!

Result Grid   Filter Rows: Export: Wrap Cell Content: ΙΑ

Field	Type	Null	Key	Default	Extra
No	int(11)	NO	PRI	NULL	auto_increment
Nama	varchar(30)	NO		NULL	
Usia	smallint(6)	YES		NULL	
Berat	float(3,1)	YES		NULL	
Kota	text	YES		NULL	
Th	year(4)	YES		NULL	
Data_In	timestamp	NO		CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP

Result will be shown here.

Join Table

Buat 2 tabel dengan 1 kolom yang beririsan,
dalam contoh kali ini yaitu kolom 'id'
(nama kolom tidak harus sama).

cabang

id	kota
1	Jakarta
2	Bandung
3	Surabaya
4	Yogyakarta

member

nama	usia	id
Andi	21	1
Budi	22	4
Caca	23	4
Dedi	24	4

Foreign Key

#1 Inner Join

The MySQL INNER JOIN clause matches rows in one table with rows in other tables and allows you to query rows that contain columns from both tables.

The following Venn diagram illustrates how the INNER JOIN clause works. The rows in the result set must appear in both tables: t1 and t2 as shown in the intersection part of two circles.

#1 Inner Join

cabang

id	kota
1	Jakarta
2	Bandung
3	Surabaya
4	Yogyakarta

member

nama	usia	id
Andi	21	1
Budi	22	4
Caca	23	4
Dedi	24	4

nama	kota
Andi	Jakarta
Budi	Yogyakarta
Caca	Yogyakarta
Dedi	Yogyakarta

```
$ SELECT nama, kota FROM member
-> INNER JOIN cabang ON
-> member.id = cabang.id;

$ SELECT nama, kota FROM member
-> JOIN cabang USING (id);
```

Purwadhika
Startup and Coding School

#2 Left Join

The MySQL LEFT JOIN clause allows you to query data from two or more database tables. The following Venn diagram helps you visualize how the LEFT JOIN clause works. The intersection between two circles are rows that match in both tables, and the remaining part of the left circle are rows in the t1 table that do not have any matching row in the t2 table. Hence, all rows in the left table are included in the result set.

#2 Left Join

cabang

id	kota
1	Jakarta
2	Bandung
3	Surabaya
4	Yogyakarta

member

nama	usia	id
Andi	21	1
Budi	22	4
Caca	23	4
Dedi	24	4

kota	nama
Jakarta	Andi
Yogyakarta	Budi
Yogyakarta	Caca
Yogyakarta	Dedi
Bandung	NULL
Surabaya	NULL

```
$ SELECT kota, nama FROM cabang
-> LEFT JOIN member USING (id);
```

Purwadhika
Startup and Coding School

View

Virtual tabel yang didefinisikan dari query select & sifatnya identik dengan tabel.

View daftar karyawan dengan BB > 60 :

```
CREATE VIEW jumbo AS
-> SELECT Nama, Berat FROM karyawan
-> WHERE Berat > 60;
```

Cek daftar tabel:

```
$ SHOW tables;
$ SHOW FULL tables;
```

Akan muncul tabel (view): jumbo

Hapus view:

```
$ DROP view jumbo;
```

Cek data dalam view:

```
$ SELECT * from jumbo;
```

View

View daftar member di Yogyakarta:

```
CREATE VIEW mem_yk AS
-> SELECT nama, kota FROM member
-> JOIN cabang USING (id)
-> WHERE kota = 'Yogyakarta';
```

Cek daftar tabel:

```
$ SHOW tables;
$ SHOW FULL tables;
```

Hapus view:

```
$ DROP view mem_yk;
```

Akan muncul tabel (view): mem_yk

Free MySQL Hosting

freemysqlhosting.net/