MySQL workbench—How to create ER diagram

Last amended: 5th Nov, 2025
My folder: D:\data\OneDrive\Documents\Database systems
MySQL Workbench manual is here

Contents

Given any relational database, here is screen-by-screen help to how to draw its ER-diagram in MySQL server Workbench. We assume you already have '*employees*' database or your database of interest already loaded in MySQL server.

(For Ubuntu OS only: For loading '*employees*' database into mysql server, please first execute the file '*er_diagram.sh*' in your virtual machine's folder: */home/ashok/Documents/erd and normalization exercises/erd_in_workbench* in the Ubuntu_database VM.)

Configuration and shortcuts:

A. All SQL Editor and Workspace bench configuration changes are saved to file:

    `C:\Users\ashok\AppData\Roaming\MySQL\Workbench\wb_options.xml`

    ```
    (A copy saved in file at folder:
    D:\OneDrive\Documents\database_systems\mysql_workbench\workbench_confi
    guration)\
    ```

B. Useful Workbench shortcut summary:

Ctrl+T	Open Query tab
Ctrl+SHIFT+O	Open sql script file
Ctrl+SHIFT+ENTER	Execute all queries
Ctrl+ENTER	Execute query in Query Editor
Ctrl+R	Reverse Engineer diablogbox
Ctrl+G	Forward Engineering
Ctrl+SHIFT+G	Write Forward Engineer code to SQL file
Ctrl+S	Save the diagram model as *.mwb
	(To import it double click on this file)
Ctrl+O	Open model (ERD) file

C. ERD diagram shortcuts summary:

Hit T and click on the workspace	Create table
Ctrl+S	Save the diagram model as *.mdb
	(To import a model, double click on this file)

Open Workbench

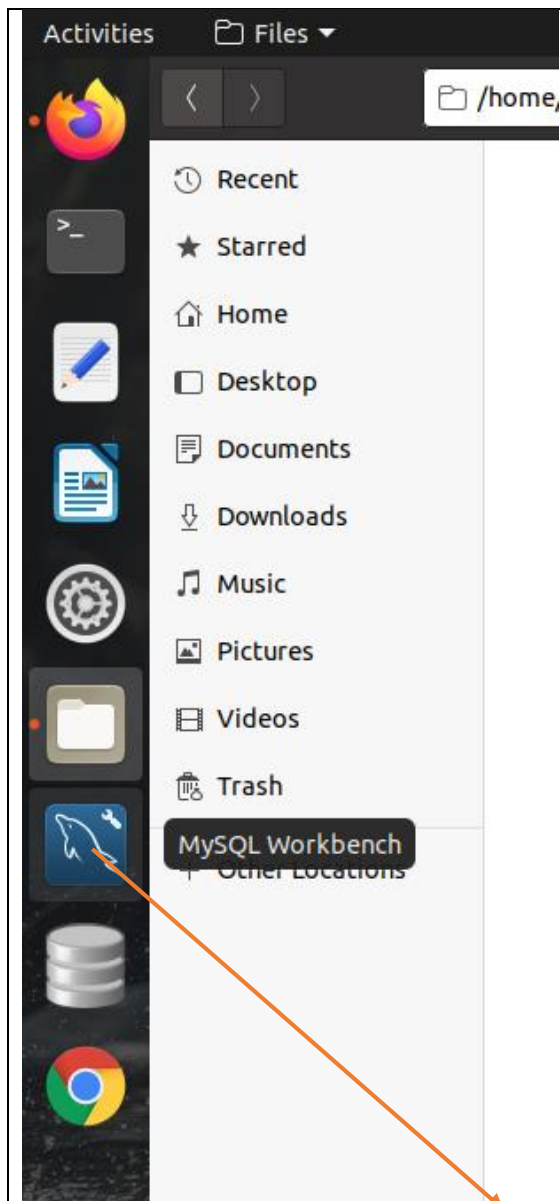In Windows use Start Menu to open MySQL Workbench (right figure). In VM, click as in left-figure

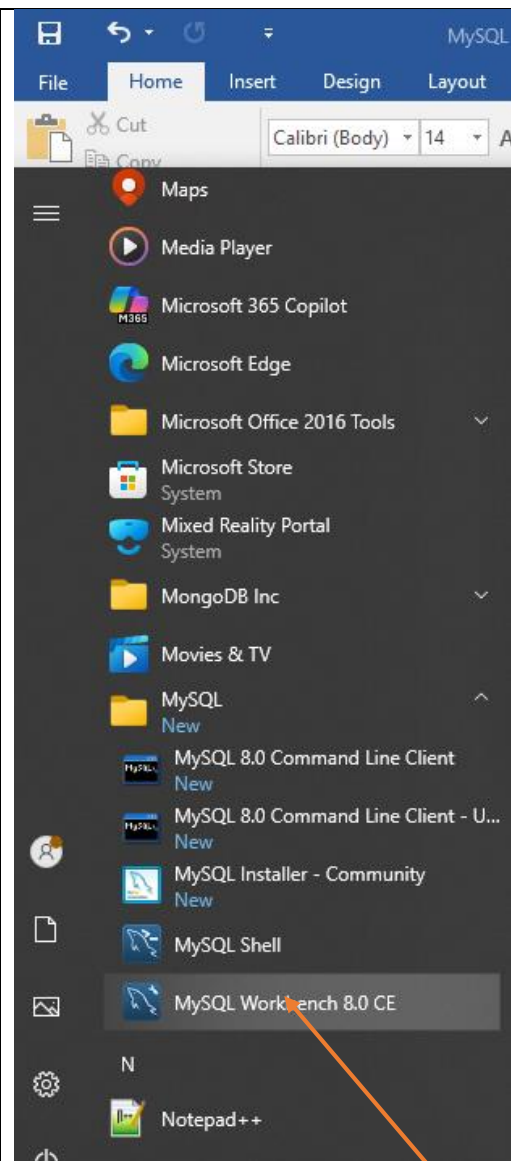Figure 1: : In Ubuntu_database VM, click on the icon of MySQL Workbench to open it.

Figure 2: From Start Menu, access MySQL-->MySQL Workbench. Better create a short-cut in the Task Bar.

When MySQL Workbench opens, click on Local Instance or (a server created by you).
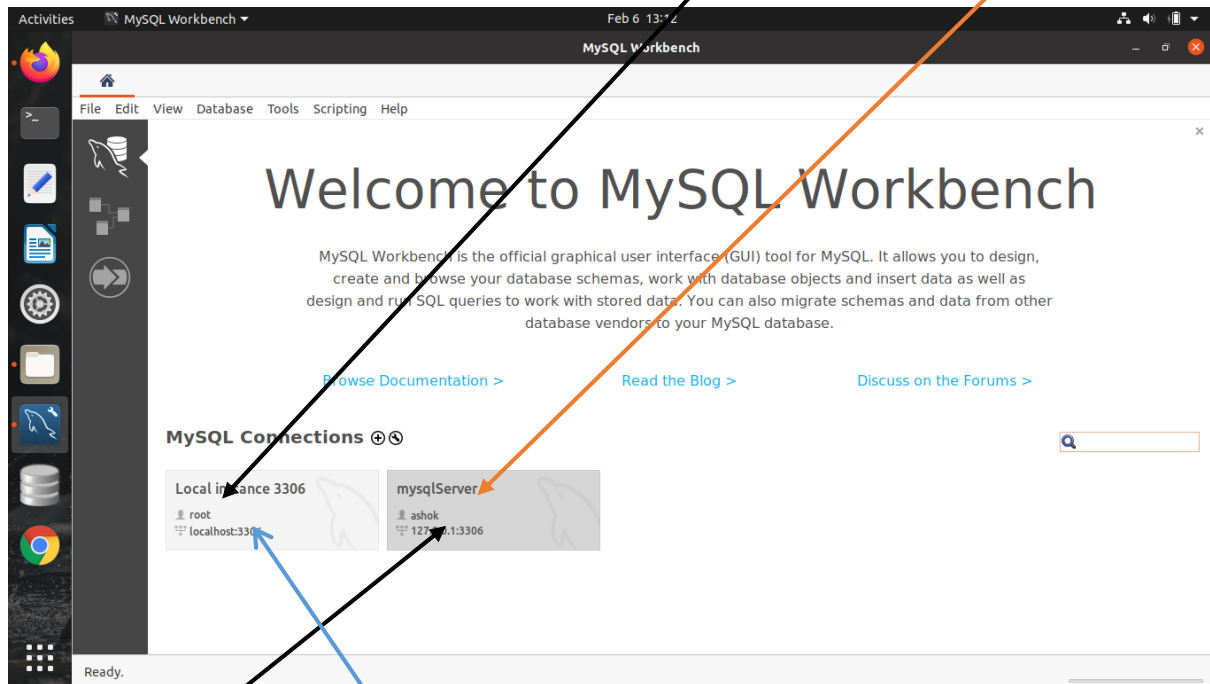

Figure 3: Click mysqlServer OR localhost link to open, as the case may be.

Workbench Configuration changes:

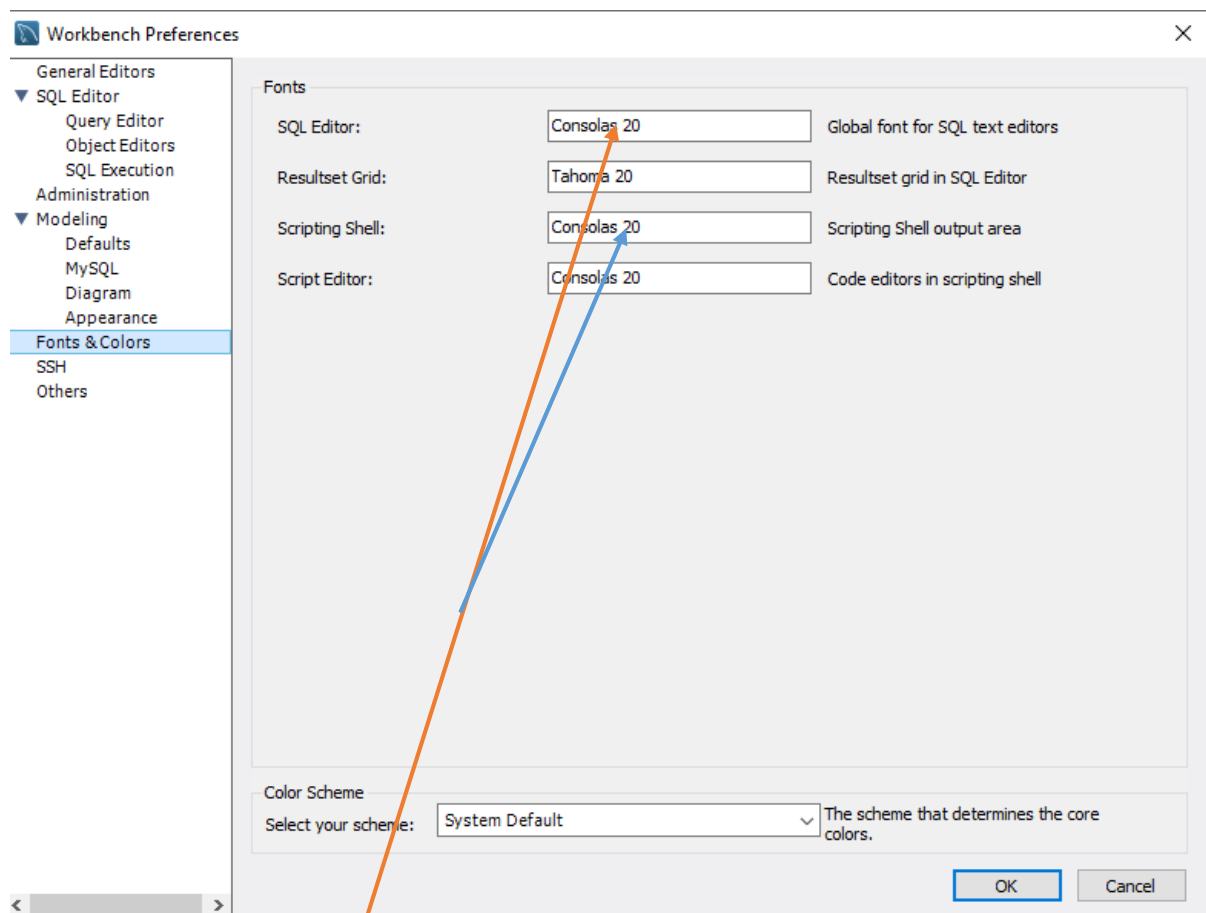Click *Edit→Preferences→Fonts and Colors*. Set everything to font size of 20. Restart the Workbench.

Figure 4: Change all fonts sizes to 20.. And restart Workbench.

Database creation

Press ctrl+T to open Query tab, if not already opened. Just create an empty database, as:
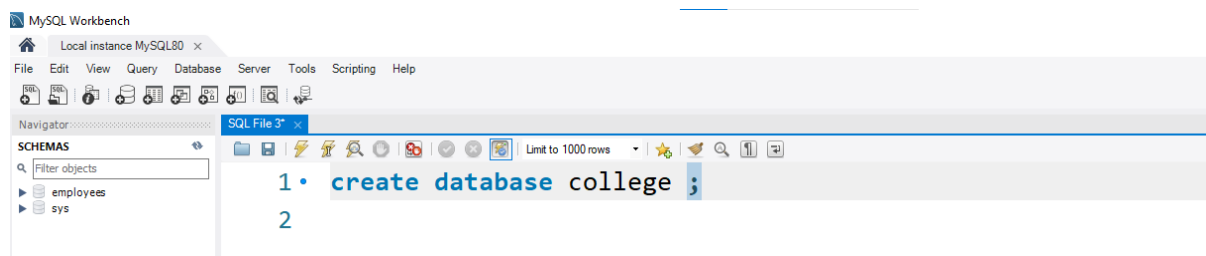
```
Create database college ;
```


Figure 5: Press ctrl+T to open a query tab. Write create statement and press ctrl+ENTER to execute the query.
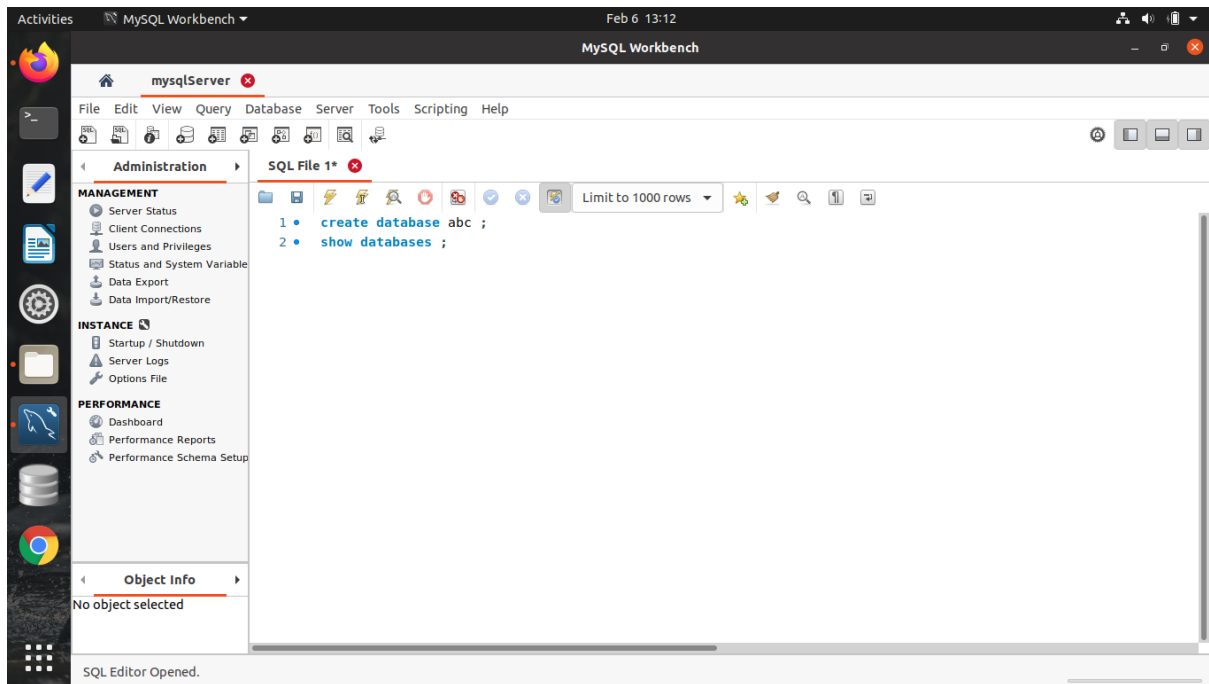
1.

*Figure 6: You will be here. Press **ctrl+R** to open another dialog box; or in the top-menu click on **Database-->Reverse Engineer**.*
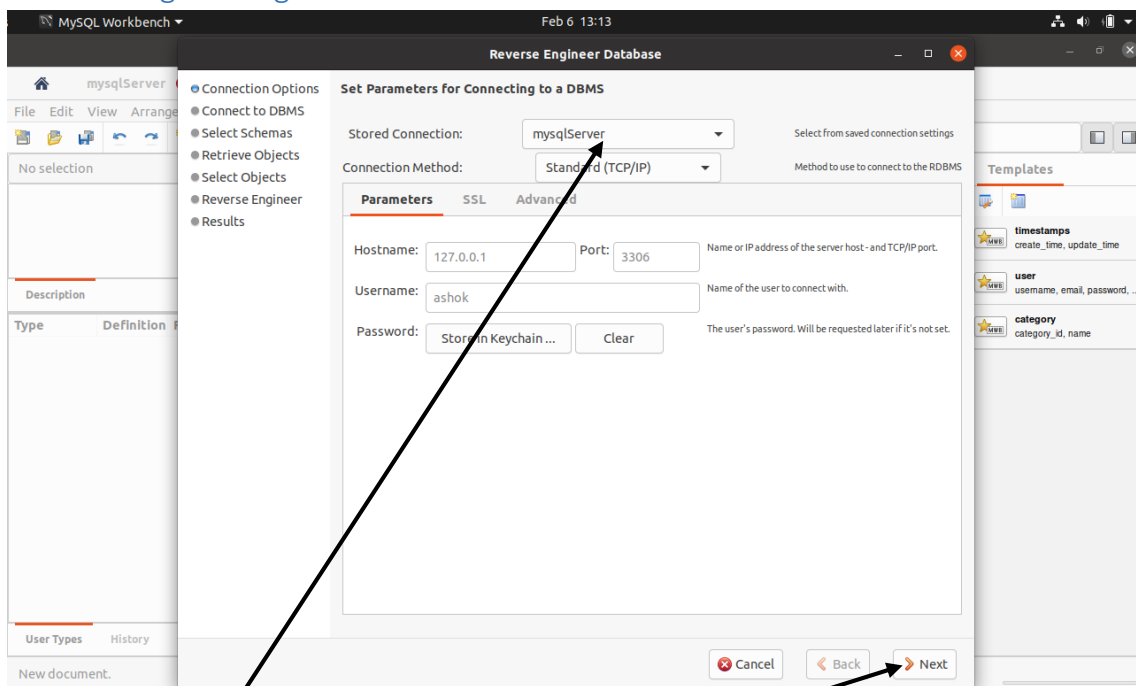
Reverse Engineering

*Figure 7: Select mysqlServer in the drop down, if not selected. Then click **Next** button.*
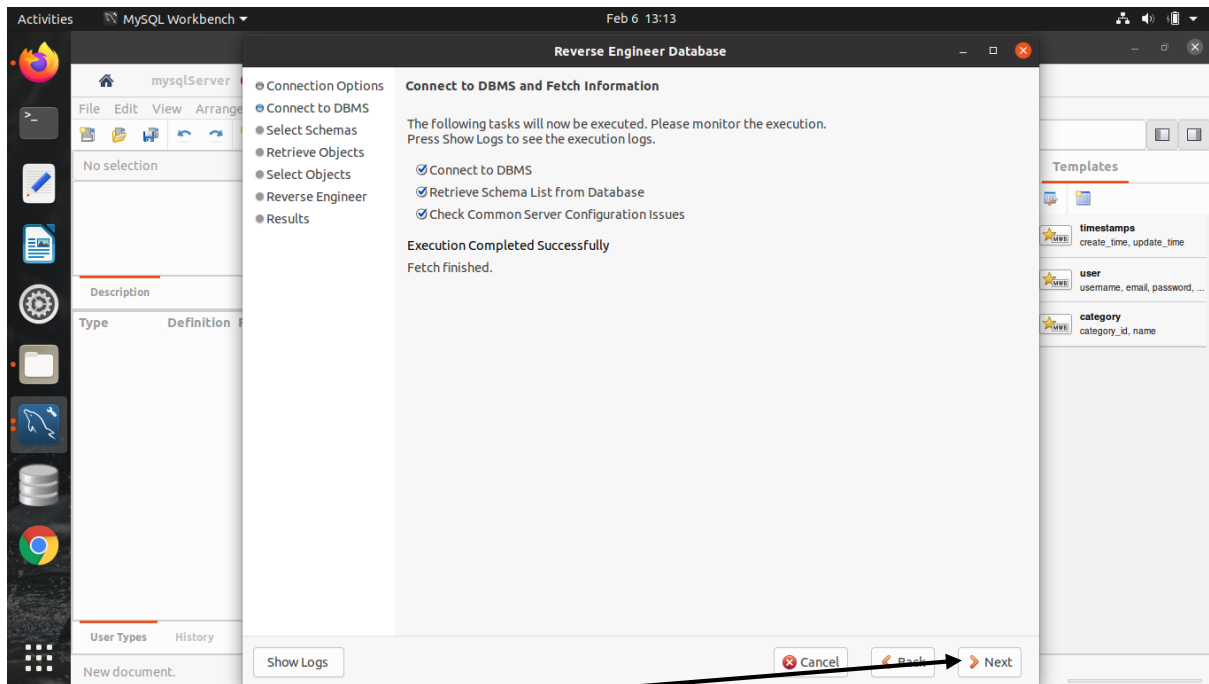
2.

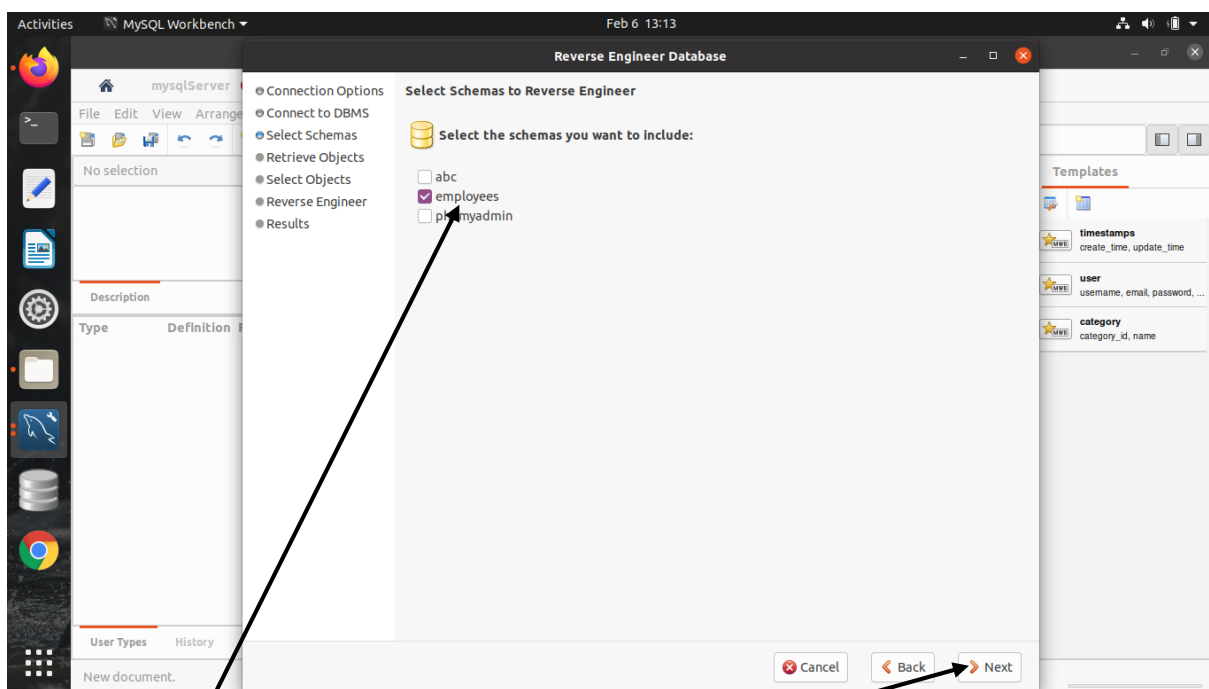*Figure 8: Nothing to do. Click **Next** button*

3.

*Figure 9: Select **'employees'** or **'college'** database or <u>your database of interest</u> and click **Next** button.*
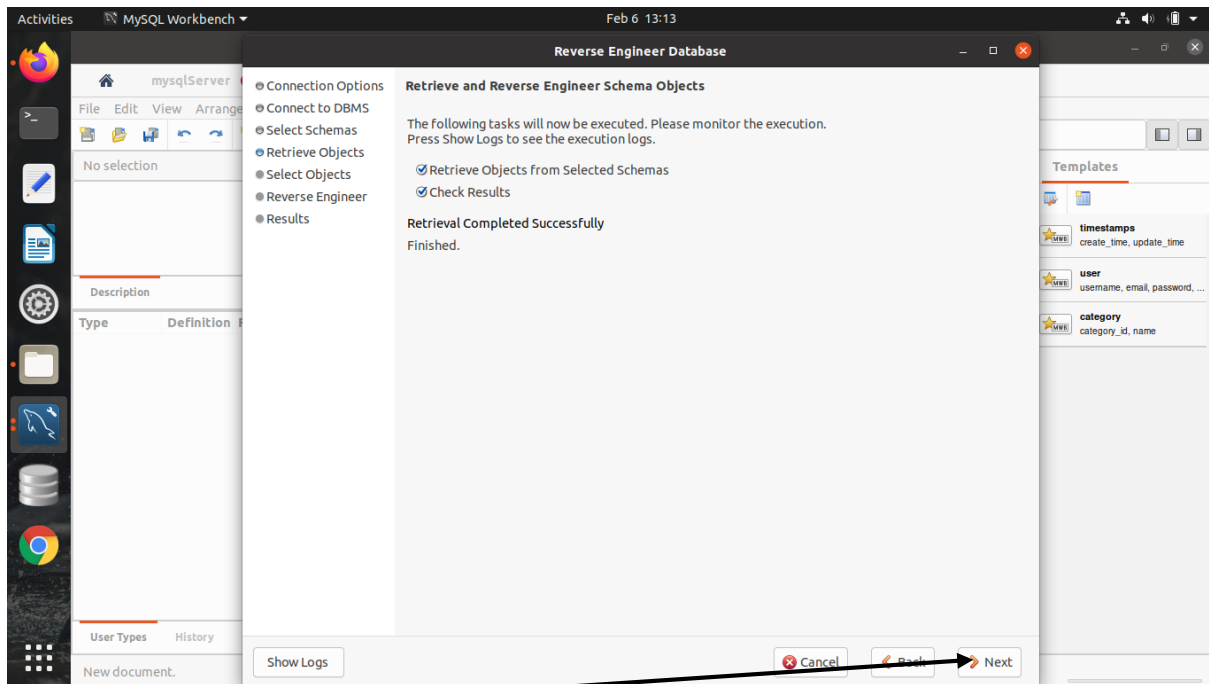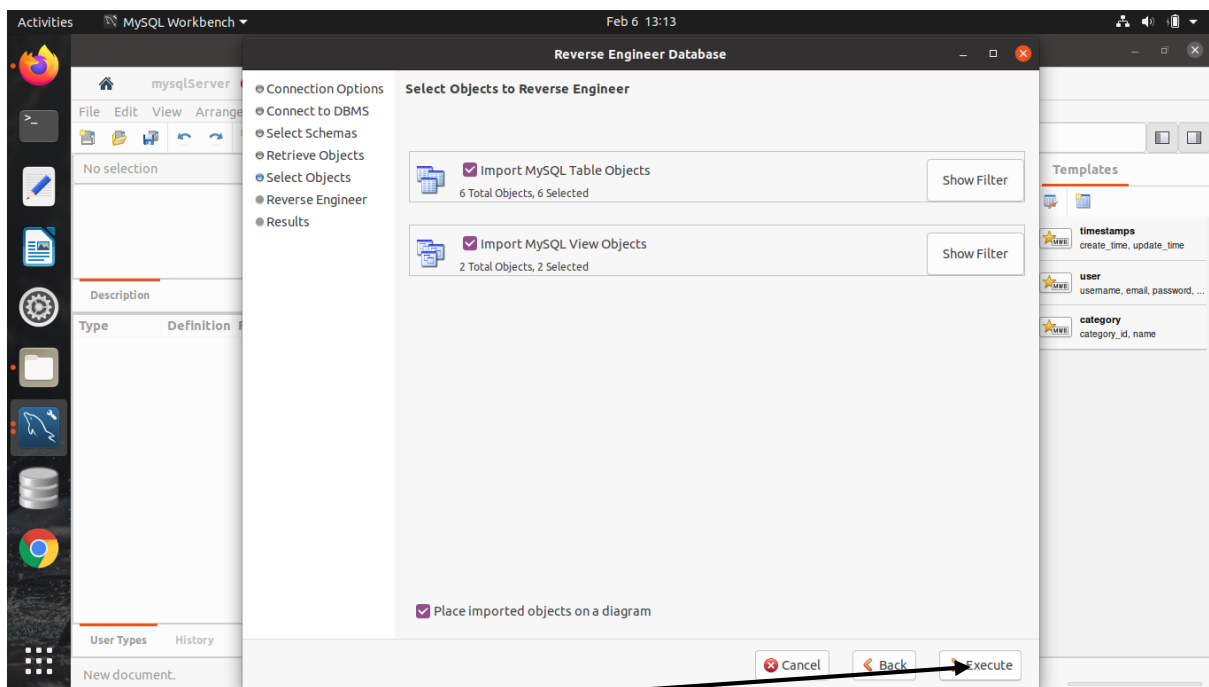
4.

*Figure 10: Click '**Next**' button.*

5.

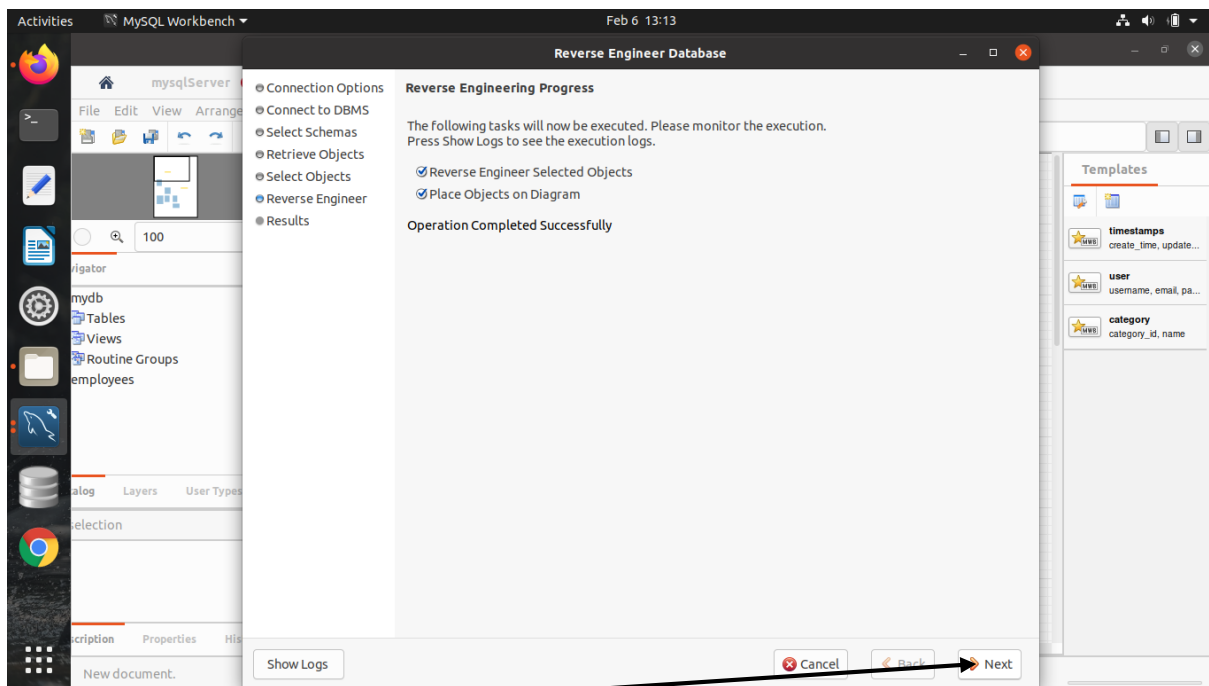*Figure 11: Click 'Execute' button*

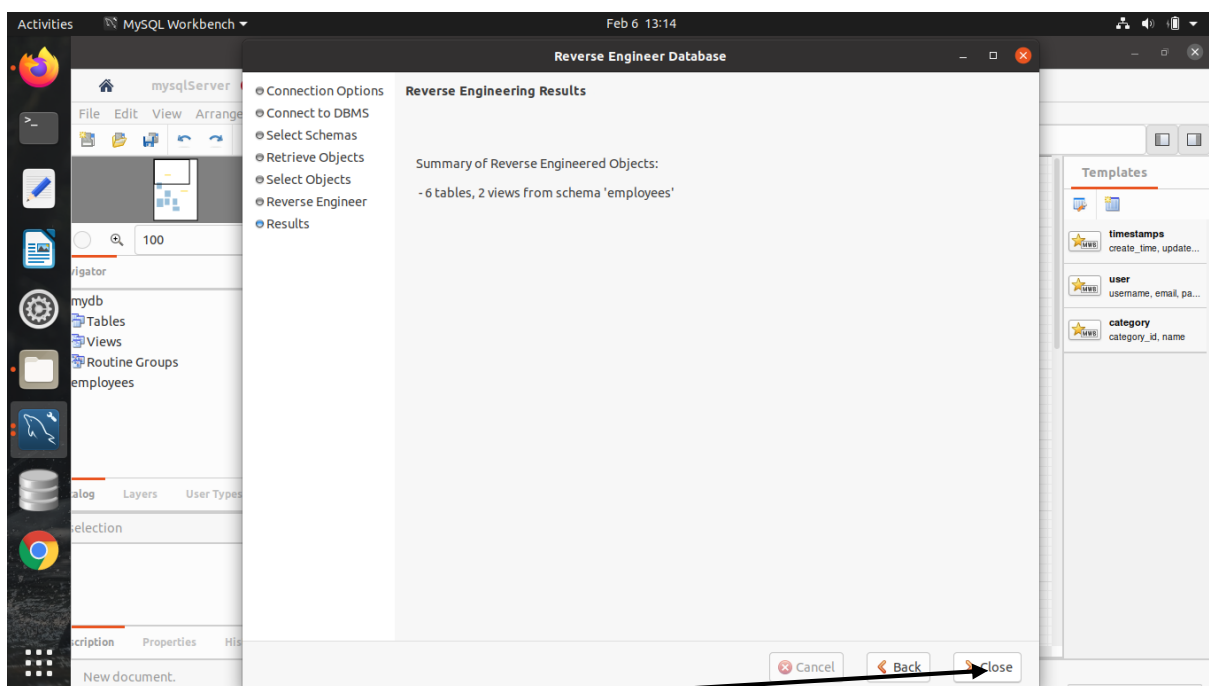**6.**

Figure 12: Click Next

7.

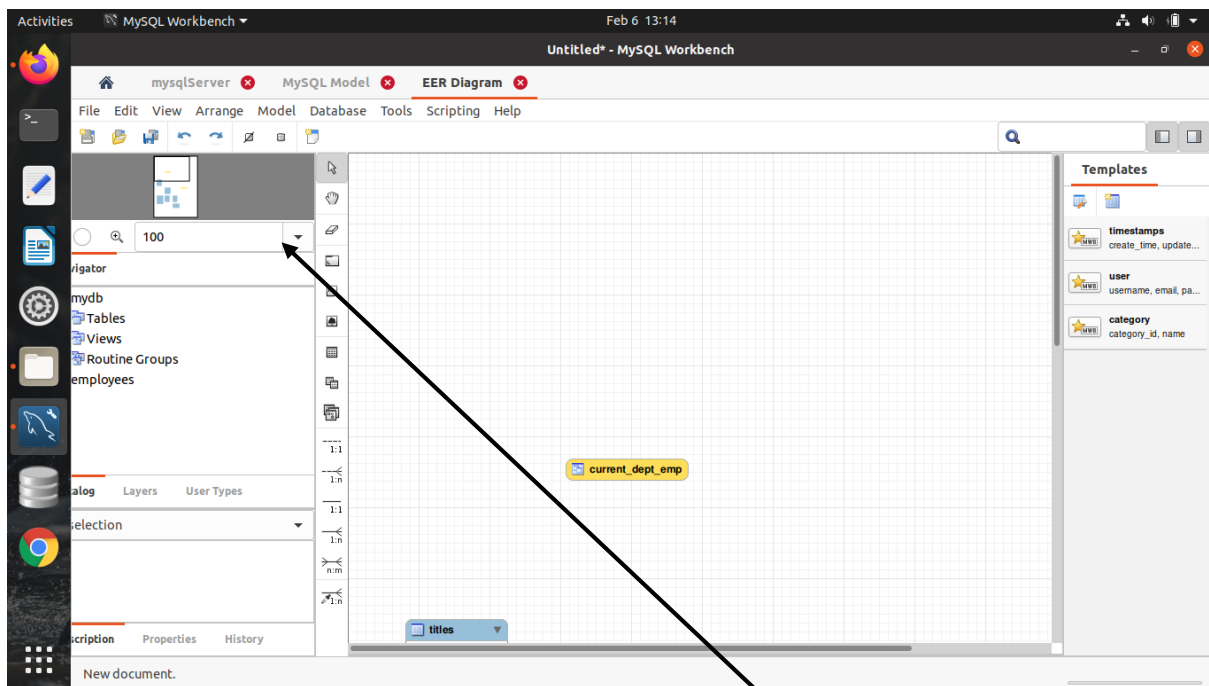

Figure 13: Click Close

8.

Figure 14: Change zoom level appropriately. In the drop-down, maybe change 100 to 75 to see er diagram.
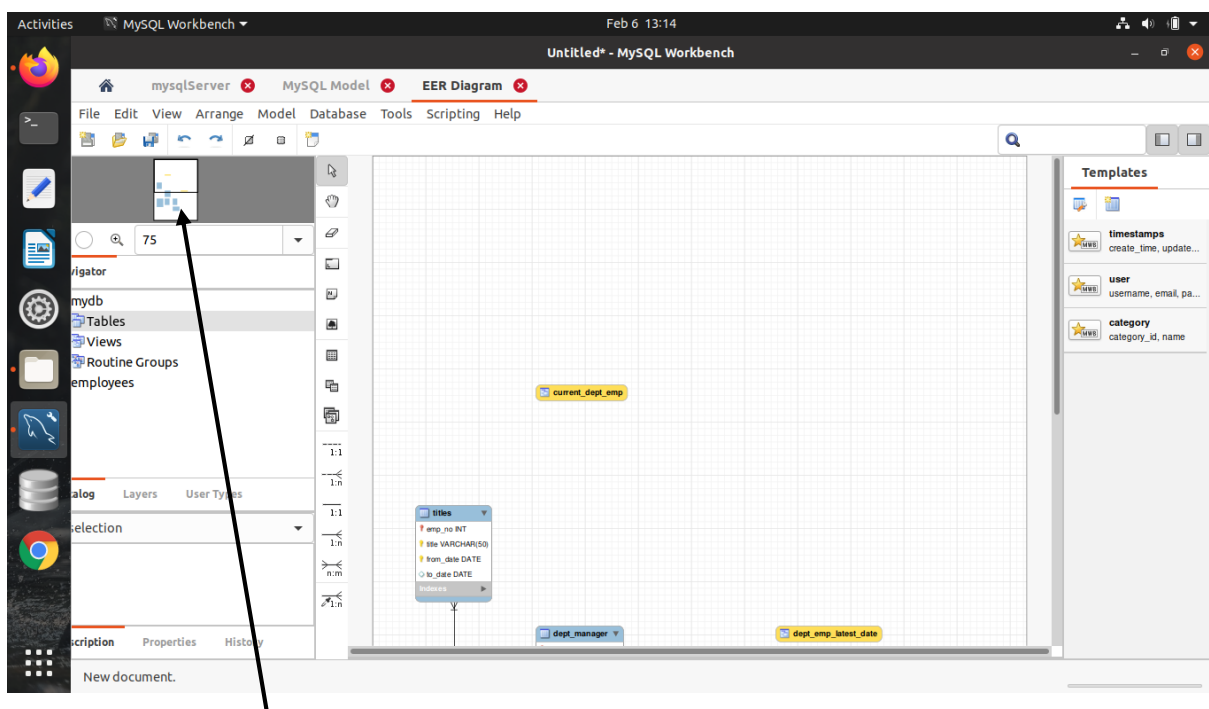
9.

Figure 15: Drag the small rectangle down so that the blue spots are within it. It is a small pre-view of your workbench.
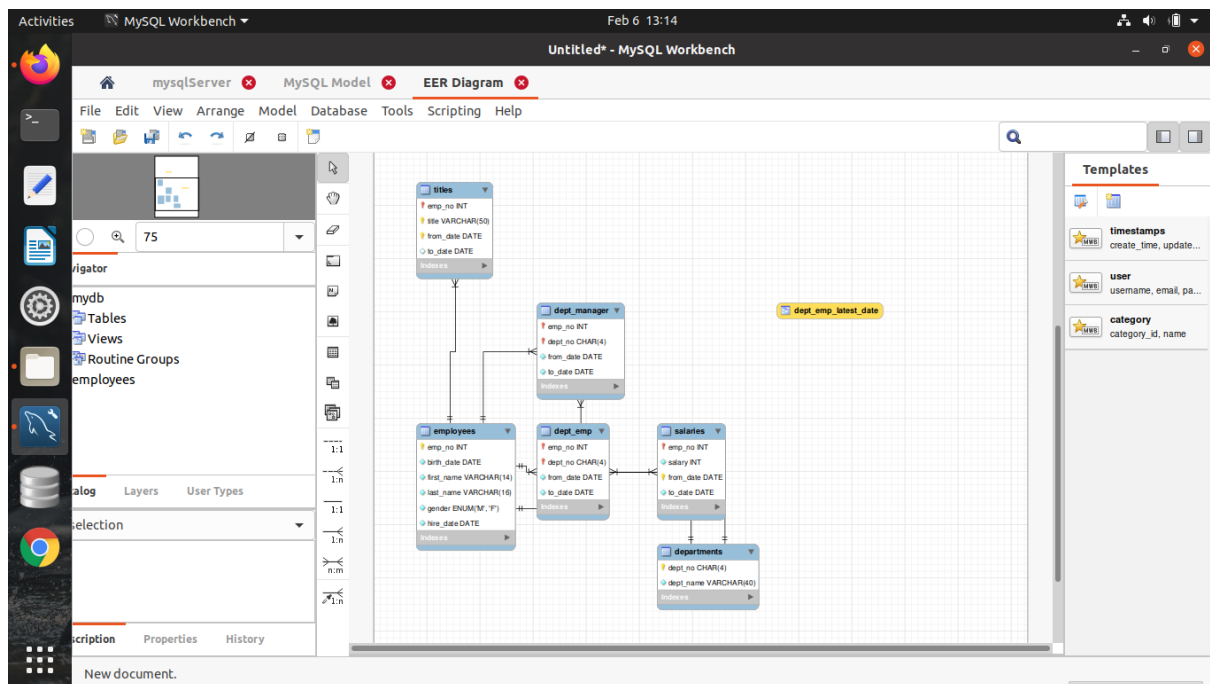
10.

Figure 16: ER diagram. Re-arrange it so that it appears nicely and lines intersect minimum possible.

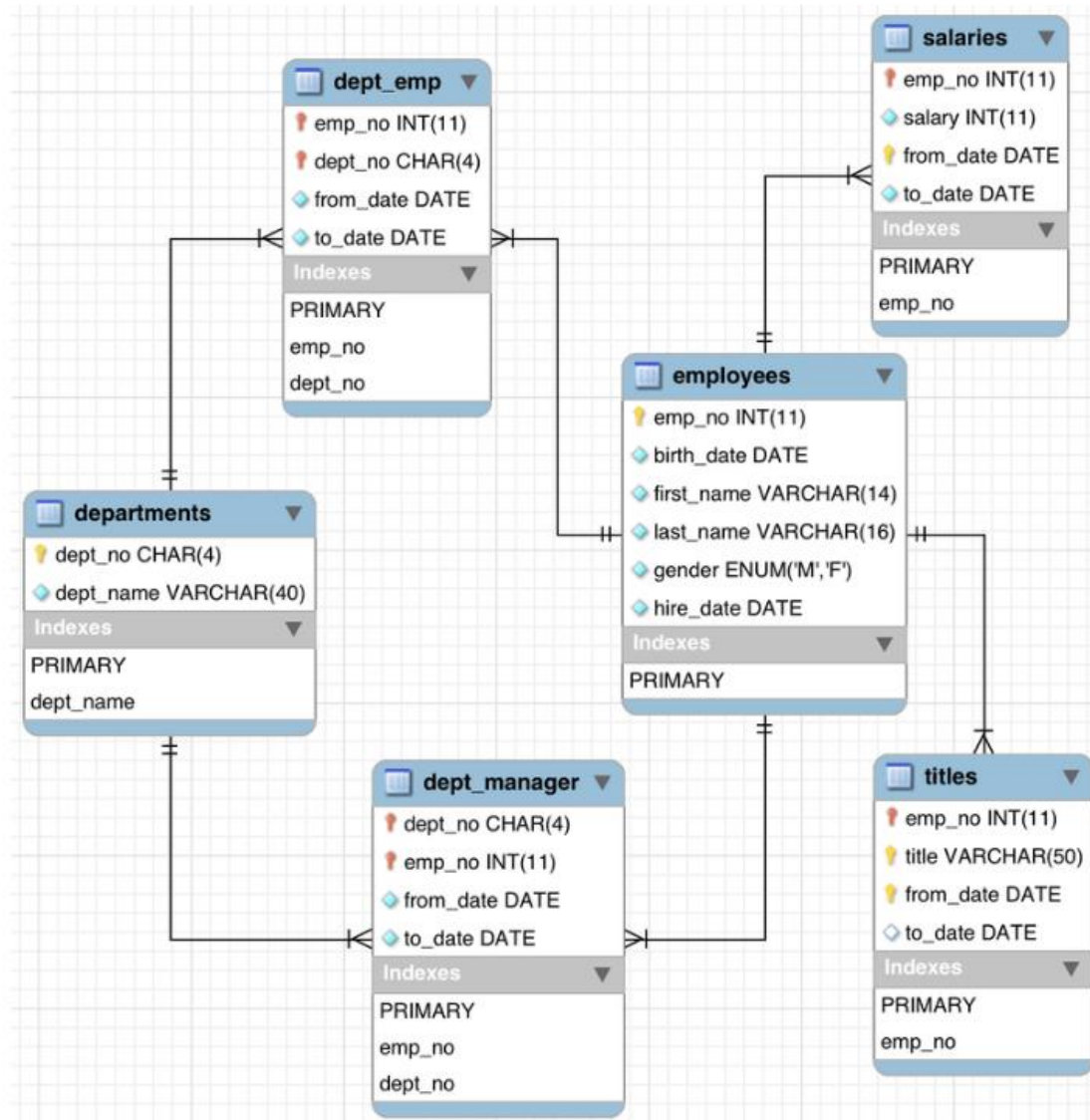See below zoomed employees database schema

Figure 17: Employees table schema—zoomed

Identifying vs non-identifying relationships

For differences among themselves, please see this link in StackOverflow. The code generated in the two cases is different. Select a relationship that suits your needs.

THERE IS NO POINT IN CREATING 1:1 RELATIONSHIP BETWEEN TWO TABLES.

For 1:n, generally use **Identifying relationship**.

Relationships between two entities may be classified as being either "identifying" or "non-identifying". Identifying relationships exist when the primary key of the parent entity is included in the primary key of the child entity. On the other hand, a non-identifying relationship exists when the primary key of the parent entity is included in the child entity but not as part of the child entity's primary key. In addition, non-identifying relationships may be further classified as being either "mandatory" or "non-mandatory". A mandatory non-identifying relationship exists when the value in the child table cannot be null. On the other hand, a non-mandatory non-identifying relationship exists when the value in the child table can be null.

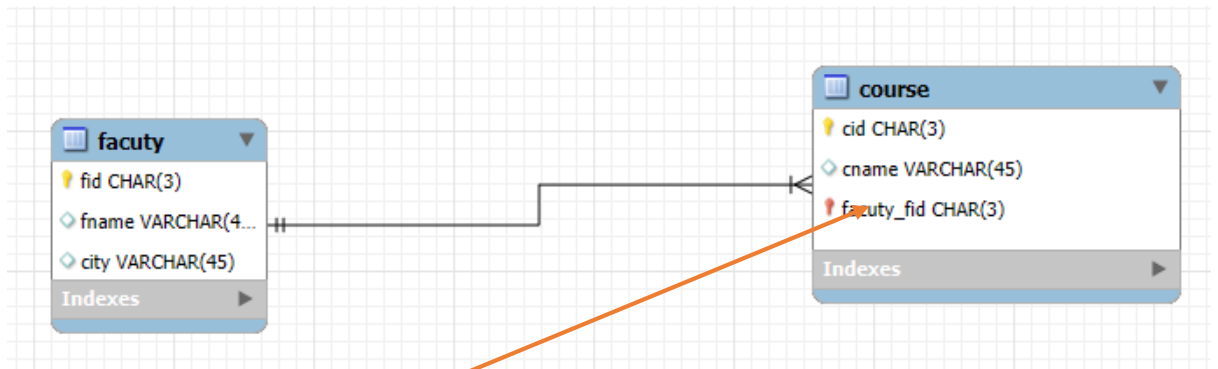*Figure 18: Dotted relationships are non-identifying. faculty_id is NOT a part of primary key in course table*

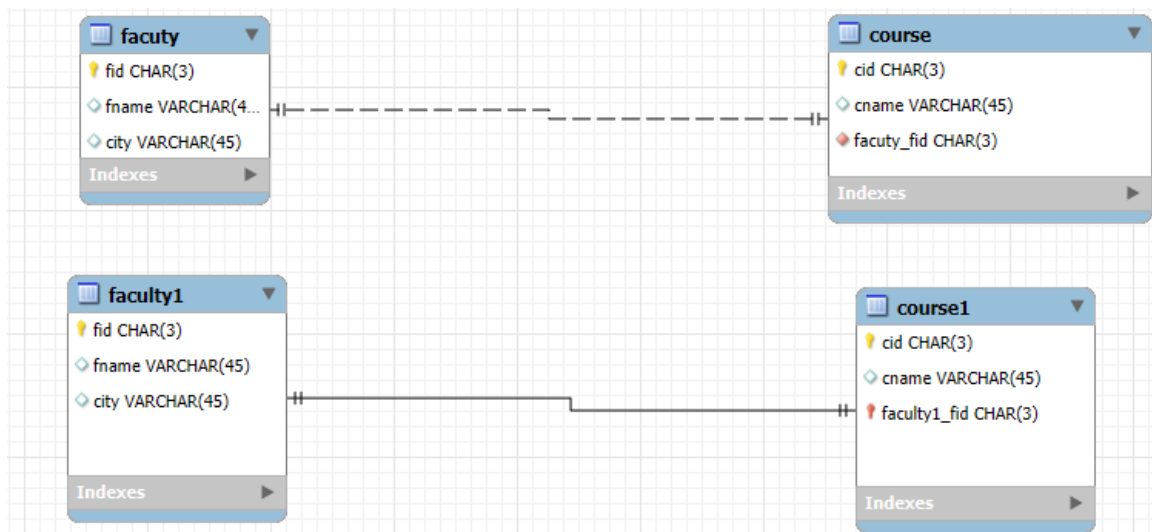*Figure 19: Identifying relationship. faculty_id IS a part of pk in course table.*

Figure 20: In the upper diagram, faculty_id is not a part of primary key in course table but in the lower diagram it is. In the upper figure, one course can be taught ONLY by one faculty. ONE-TO-ONE RELATIONSHIPS ARE MEANINGLESS.
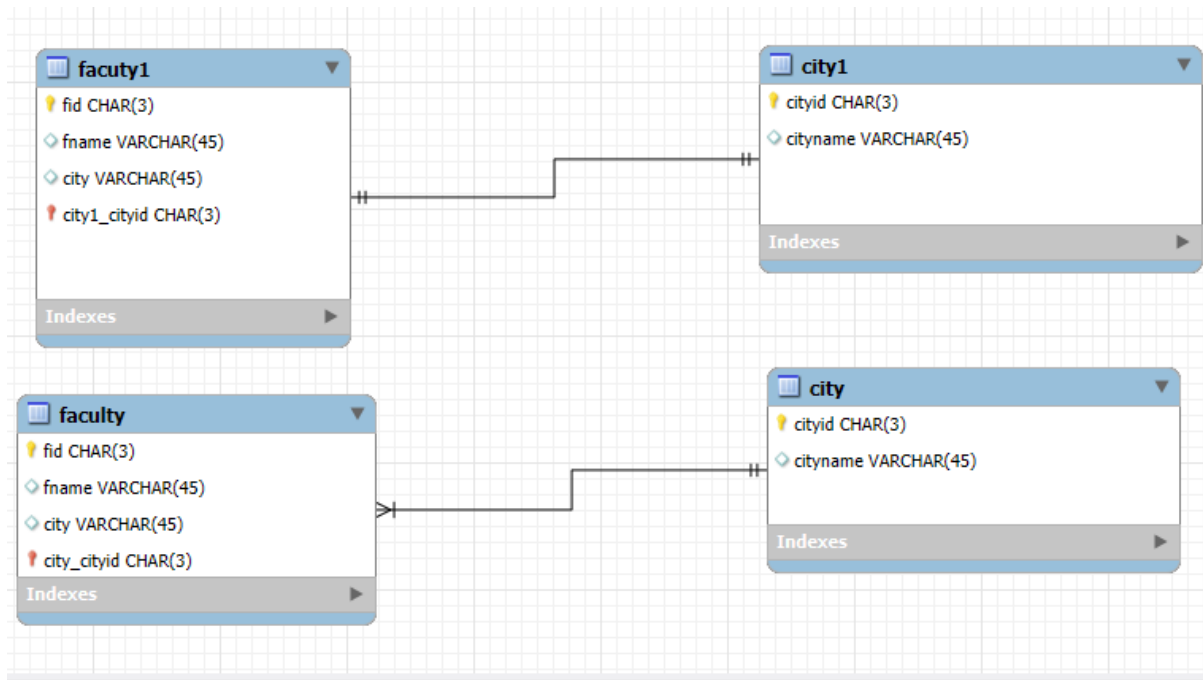
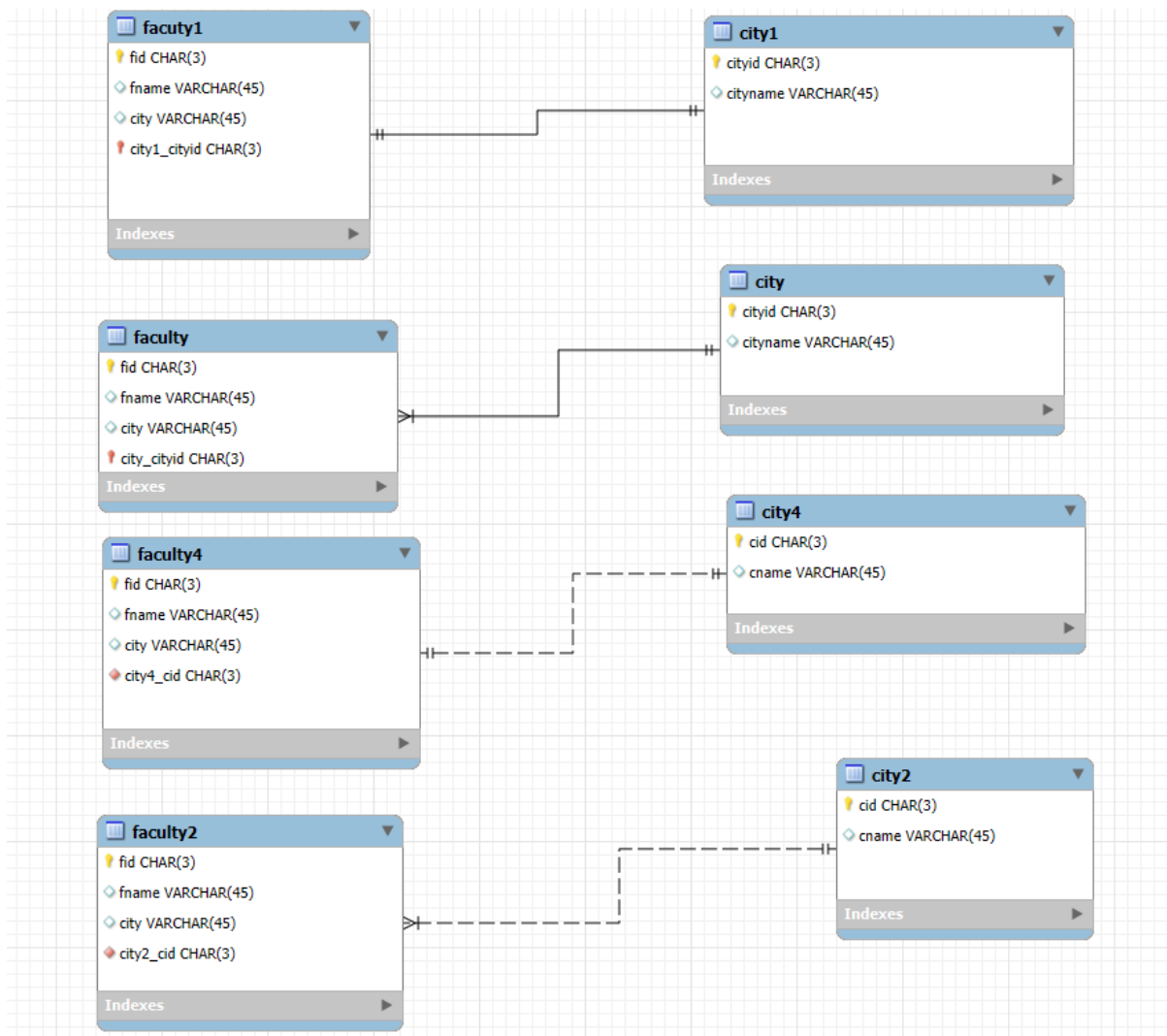*Figure 21: Note that both the above diagrams display the same relationships*

Figure 22: A comparison of different relationships

Faculty-Courses-city ERD

In the following ERD, faculty is the most important entity followed by course. Yet it is the city's table that must be filled up first. Note that 'city' is now redundant in faculty table.
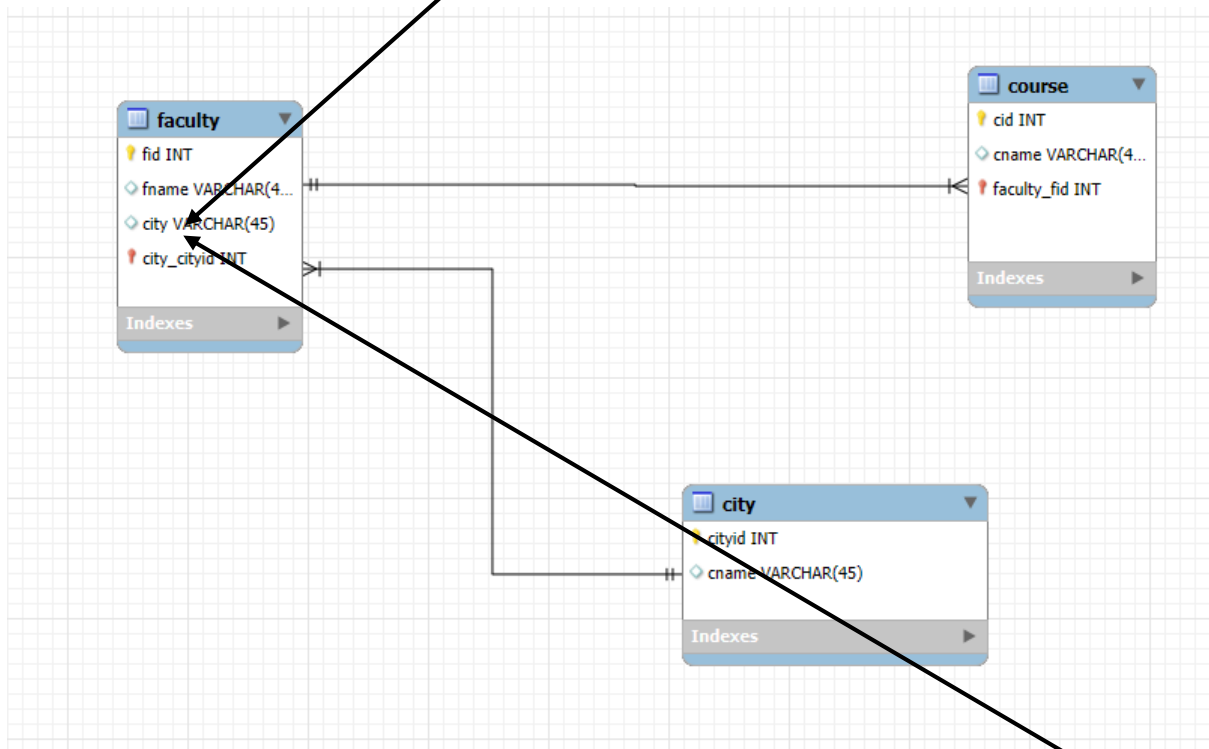

Figure 23:While faculty is the most important entity here, it is the city's table that must be filled up first. Note that city field is now redundant in faculty table.

Forward Engineering

Forward Engineering: Press ctrl+G to perform Forward Engineering. And press ctrl+SHIFT+G to save forward engineered script.

Restart Workbench to find the database changed.

Row insertion

To insert a row in any table.

> under schemas,
> > click your database, say, college,
> > > right click on a table, say city,
> > > > then click on **Send to SQL Editor→Insert statement**
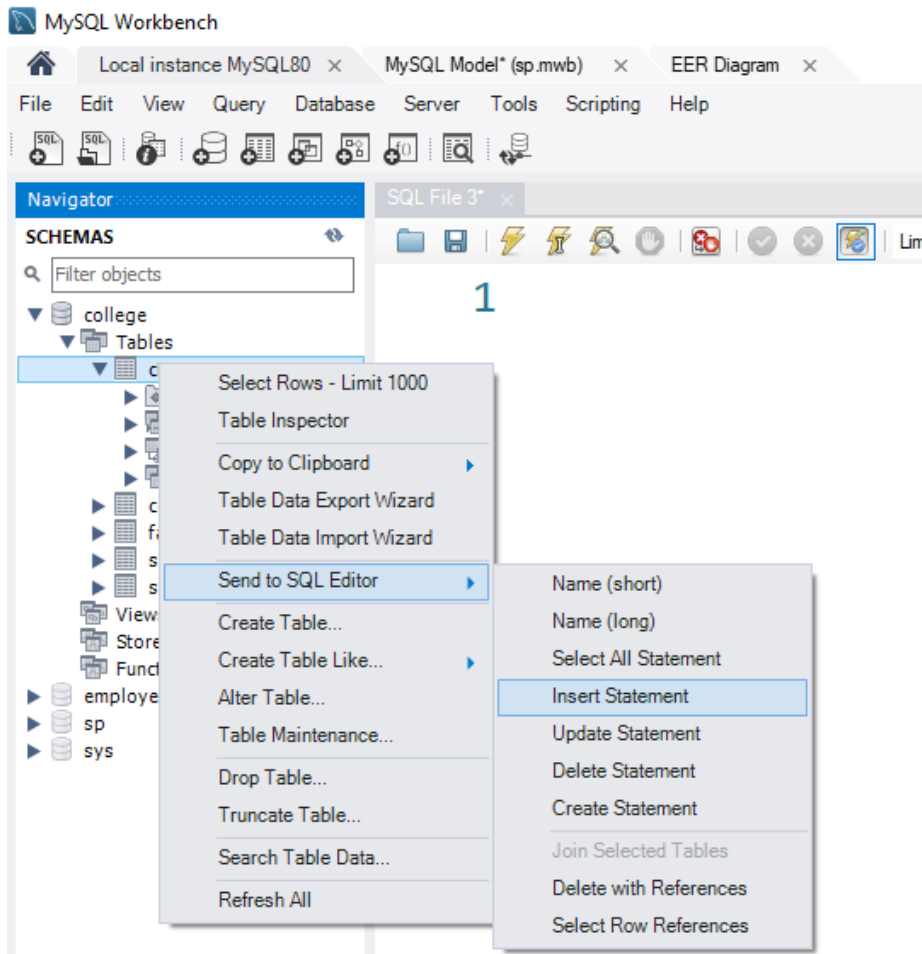
(See figure below)

Figure 24: Generating an insert statement in SQL editor for a table

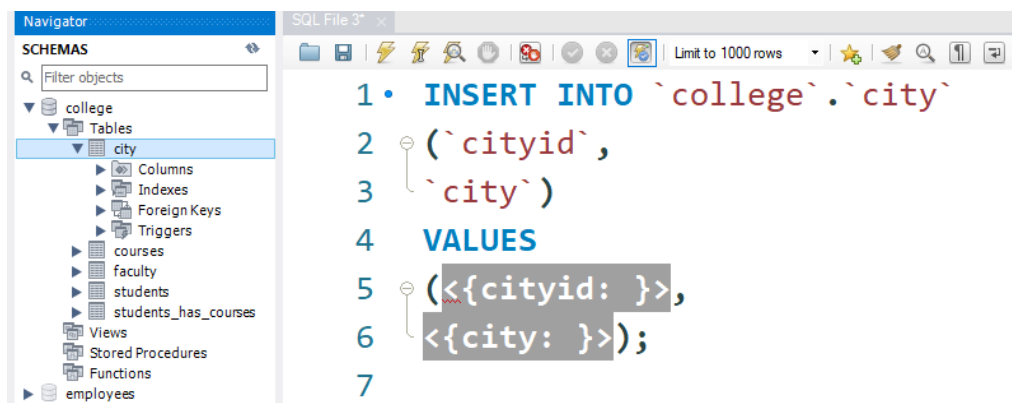An insert statement appears, fill in values and execute:

Figure 25: Replace $<\{cityid: \}>$ and <{city:}> with actual values in insert statement and execute

MySQL Data types

MySQL offers a comprehensive range of data types categorized into numeric, string, and date/time types, along with specialized types like spatial and JSON.

1. Numeric Data Types:

- **Integers:**

Used for whole numbers.

- o `TINYINT`: Very small integers (-128 to 127 signed, 0 to 255 unsigned). Example: `age TINYINT UNSIGNED`.
- o `SMALLINT`: Small integers (-32768 to 32767 signed). Example: `postal_code SMALLINT`.
- o `MEDIUMINT`: Medium-sized integers. Example: `employee_id MEDIUMINT`.
- o `INT` (or `INTEGER`): Standard integers. Example: `user_count INT`.
- o `BIGINT`: Large integers. Example: `population BIGINT`.
- **Floating-Point Numbers:**

Used for numbers with decimal points.

- o `FLOAT(p)`: Single-precision floating-point number. Example: `price FLOAT(10,2)`.
- o `DOUBLE(p,s)`: Double-precision floating-point number. Example: `latitude DOUBLE(9,6)`.
- **Fixed-Point Numbers:**
  - o `DECIMAL(p,s)` (or `NUMERIC`): Exact decimal representation, where `p` is the total number of digits and `s` is the number of digits after the decimal point. Example: `salary DECIMAL(10,2)`.
- **Boolean:**
  - o `BOOLEAN` (or `BOOL`): A synonym for `TINYINT(1)`, representing true (1) or false (0). Example: `is_active BOOLEAN`.

2. String Data Types:

- **Fixed-Length Strings:**
  - o `CHAR(size)`: Stores a fixed-length string, padded with spaces if shorter than `size`. Example: `country_code CHAR(2)`.
- **Variable-Length Strings:**
  - o `VARCHAR(size)`: Stores a variable-length string, up to `size` characters. Example: `product_name VARCHAR(255)`.
- **Large Text:**
  - o `TINYTEXT`, `TEXT`, `MEDIUMTEXT`, `LONGTEXT`: Used for storing large amounts of text data, with increasing storage capacity. Example: `product_description TEXT`.
- **Binary Strings:**
  - o `BINARY(size)`, `VARBINARY(size)`: Similar to `CHAR` and `VARCHAR` but store binary byte strings. Example: `checksum VARBINARY(32)`.
- **Binary Large Objects:**
  - o `TINYBLOB`, `BLOB`, `MEDIUMBLOB`, `LONGBLOB`: Used for storing binary data like images or files. Example: `profile_picture BLOB`.
- **Enumerated Type:**
  - o `ENUM('value1', 'value2', ...)`: Allows a column to have one value from a predefined list. Example: `status ENUM('active', 'inactive', 'pending')`.

3. Date and Time Data Types:

- `DATE`: Stores a date in 'YYYY-MM-DD' format. Example: `birth_date DATE`.
- `TIME`: Stores a time in 'HH:MM:SS' format. Example: `start_time TIME`.
- `DATETIME`: Stores both date and time in 'YYYY-MM-DD HH:MM:SS' format. Example: `event_timestamp DATETIME`.
- `TIMESTAMP`: Stores a timestamp, automatically updated on record modification by default. Example: `last_updated TIMESTAMP`.
- `YEAR`: Stores a year in 2-digit or 4-digit format. Example: `release_year YEAR(4)`.

Examples:

```
-- Some Basic experiments in databases

-- 1. Remove database(s)
drop database if exists college ;
-- 1.1
drop database if exists univ ;

-- 2. Create databases
create database if not exists college ;
-- 2.1
create database if not exists univ ;

-- 3. Create tables
use college ;
-- 3.1
drop table if exists student ;

-- 4. create a very simple table
create table faculty ( fid int , fname varchar(40) ) ;

-- 5. Execute the following mutiple times.
insert into faculty values ( 123 , 'abc' ) ;
-- 5.1 Duplicate insertion gets permitted just as in Excel
insert into faculty values ( 123 , 'abc' ) ;
-- 5.2 This fails, unlike excel
insert into faculty values ('def' , 'cde') ;

-- 6. Check this table disallows duplicate fid, unlike excel
create table faculty ( fid int primary key,
                       fname varchar(40) ) ;

-- 7. More constraints
drop table faculty ;

-- 8. Check this table disallows age < 24
create table faculty ( fid int primary key,
                       fname varchar(45) ,
                       age float check (age > 24)
                       ) ;

-- 8.1
insert into faculty values (123, 'abc', 25 ) ;
-- 8.2 This fails
insert into faculty values (123, 'abc', 23 ) ;

-- *******************--
```

MySQL User creation

Create a user 'ravi' with password, ashok. User, 'ravi' can access database only from localhost.

```
-- In SQL editor Create a user who can access from localhost
create user 'ravi'@'localhost' identified by 'ashok' ;
```

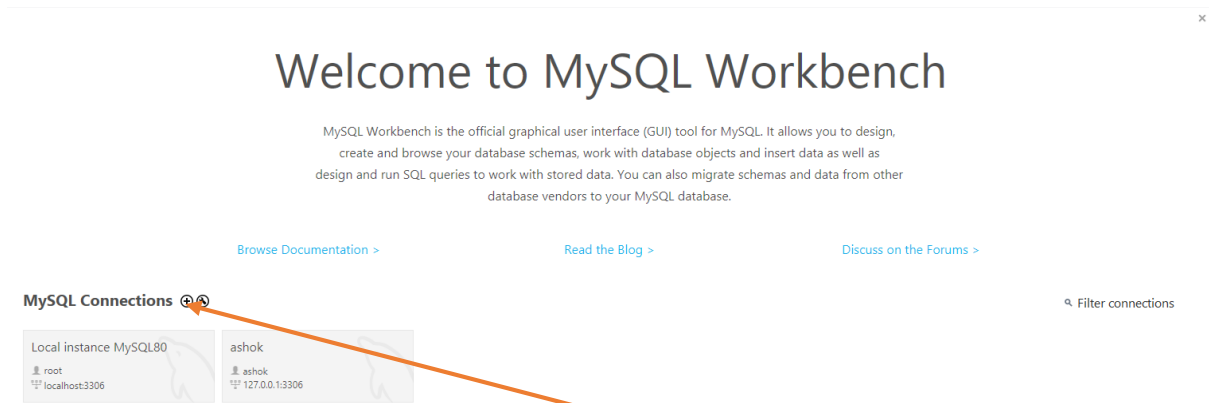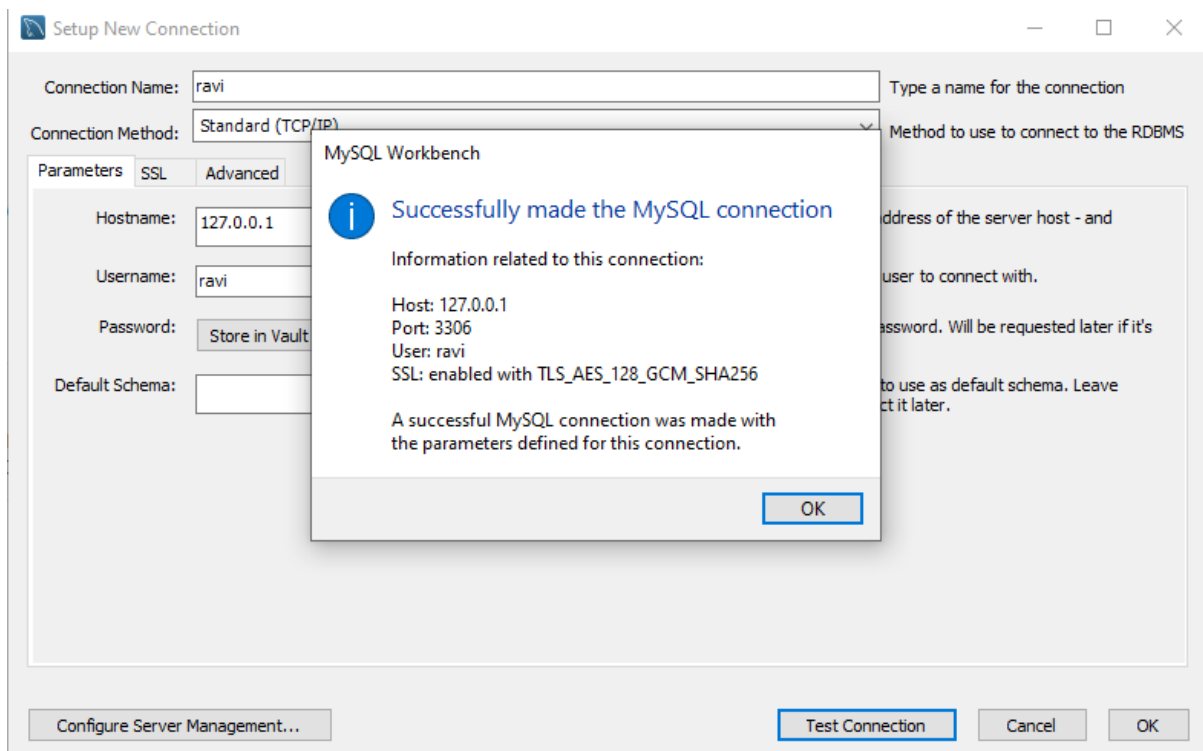■ Close MySQL Workbench and follow the steps below:

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 MySQL80   ashok
  👤 root   👤 ashok
  🖥 localhost:3306   🖥 127.0.0.1:3306

Figure 26: We have here only two users. One is root and the other is 'ashok' . Click on + icon here to create a new connection with another user.

Setup New Connection   —   □   ✕

Connection Name: ra|

Connection Method: Standard (TCP/IP)   ▼   Method to use to connect to the RDBMS

Parameters   SSL   Advanced

Hostname: 127.0.0.1   Port: 3306   Name or IP address of the server host - and TCP/IP port.

Username: ravi| Name of the user to connect with.

Password: Store in Vault ...   Clear   The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

Configure Server Management...   Test Connection   Cancel   OK

Figure 27: Fill up two information: connection name and user name.
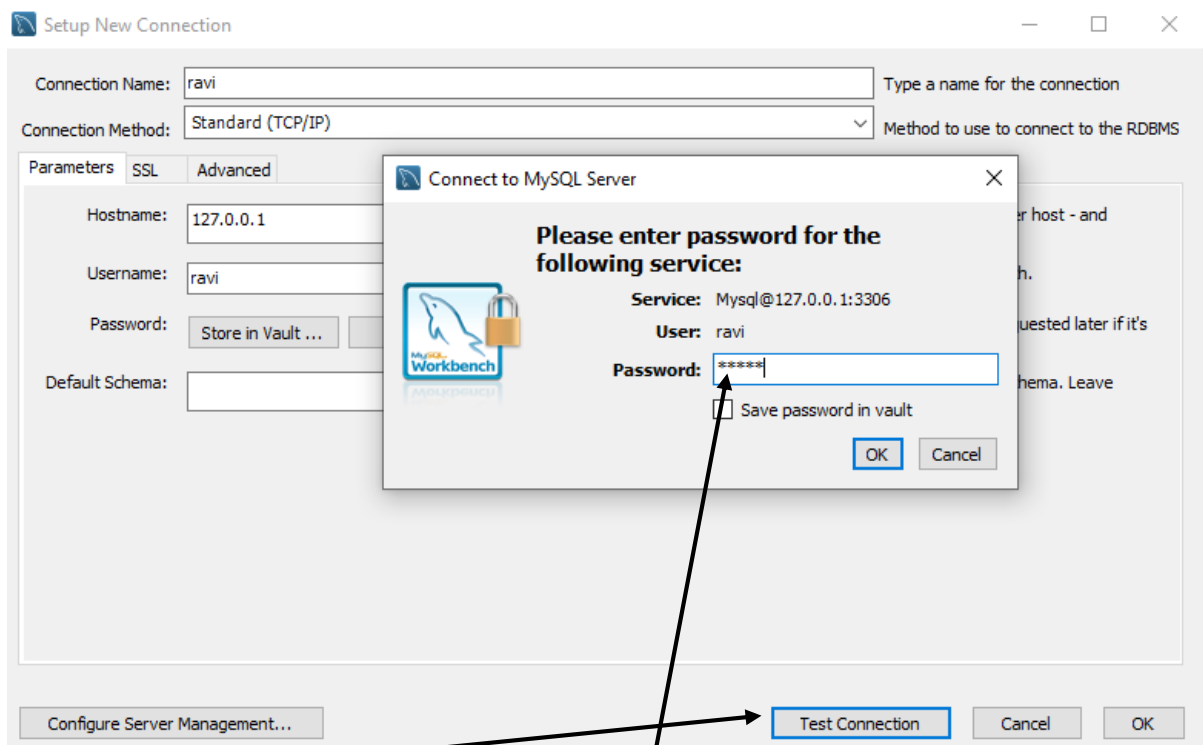
Figure 28: Click Test Connection to test the connection. Fill up password and it should succeed.

Figure 29: Connection succeeds

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 ⊕ ⊗

Local instance MySQL80	ashok	ravi
👤 root	👤 ashok	👤 ravi
🖥 localhost:3306	🖥 127.0.0.1:3306	🖥 127.0.0.1:330
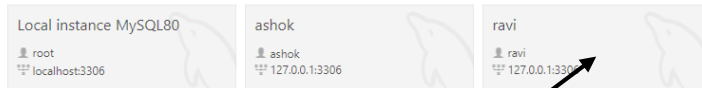
Figure 30: A new connection by name of ravi appears. Open it.

The following sql command fails for user 'ravi'.

```
create database college ;
```

Close MySQL Workbench, open again as root. And issue the following SQL command:

```
GRANT SELECT ON sakila.* TO 'ravi'@'localhost';
```

Close MySQL Workbench. Open it as user `ravi`. Issue the following two commands:

```
use sakila ;
```

```
select * from actor ;
```

See the output in the figure below.
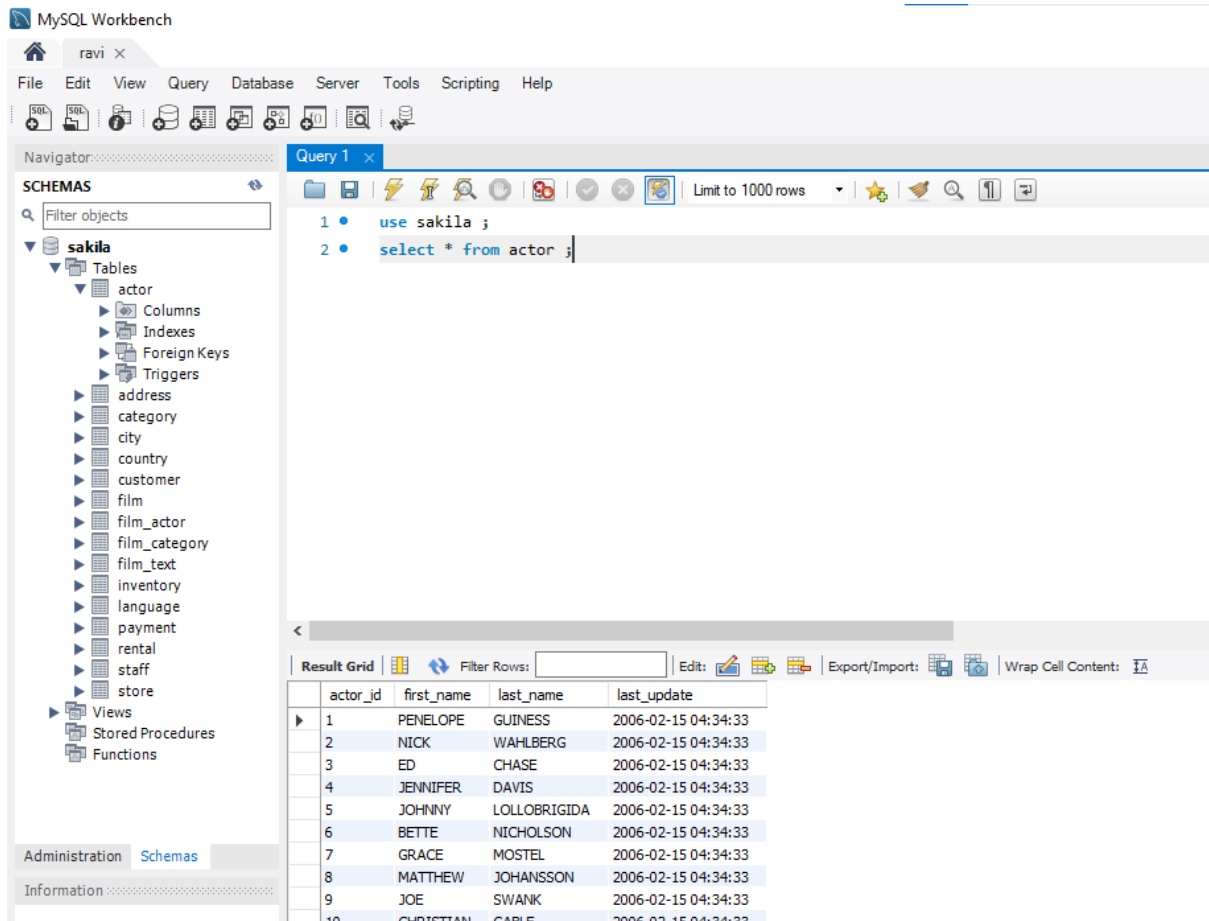
MySQL shell

To access *mysql sh*ell (`mysqlsh`), open command prompt and enter the `mysqlsh` command, as below:

Figure 31: Open command prompt and connect to mysql shell

Figure 32: Enter sql mode by issue of \sql.

Figure 33: Then issue show databases command or show tables comands

*Figure 34: Three commands: i) use college ,ii) show tables iii) desc faculty commands. You can also issue select * from..*

Grant Experiments

Open MySQL server-bench as `root` user and open command prompt. In Mysql server bench, create a user `gautam`, as:

```
create user 'gautam'@'localhost' identified by 'gautam' ;
```

In command prompt, log into `mysqlsh` as user `gautam`. Issue `show databases` command. No user databases are visible. Then in the MySQL Server Bench, issue the following grant command:

```
grant select on college.faculty to 'gautam'@'localhost' ;

(as against:

GRANT CREATE ON college.* TO gautam@'localhost' ;)
```

Figure 35: User gautam's authorization in steps. The second show databases command is after grant statement by root.

Figure 36: Affect of grant statement. First show databases is when no grant is issued.

You can also connect to root in mysql shell, as:
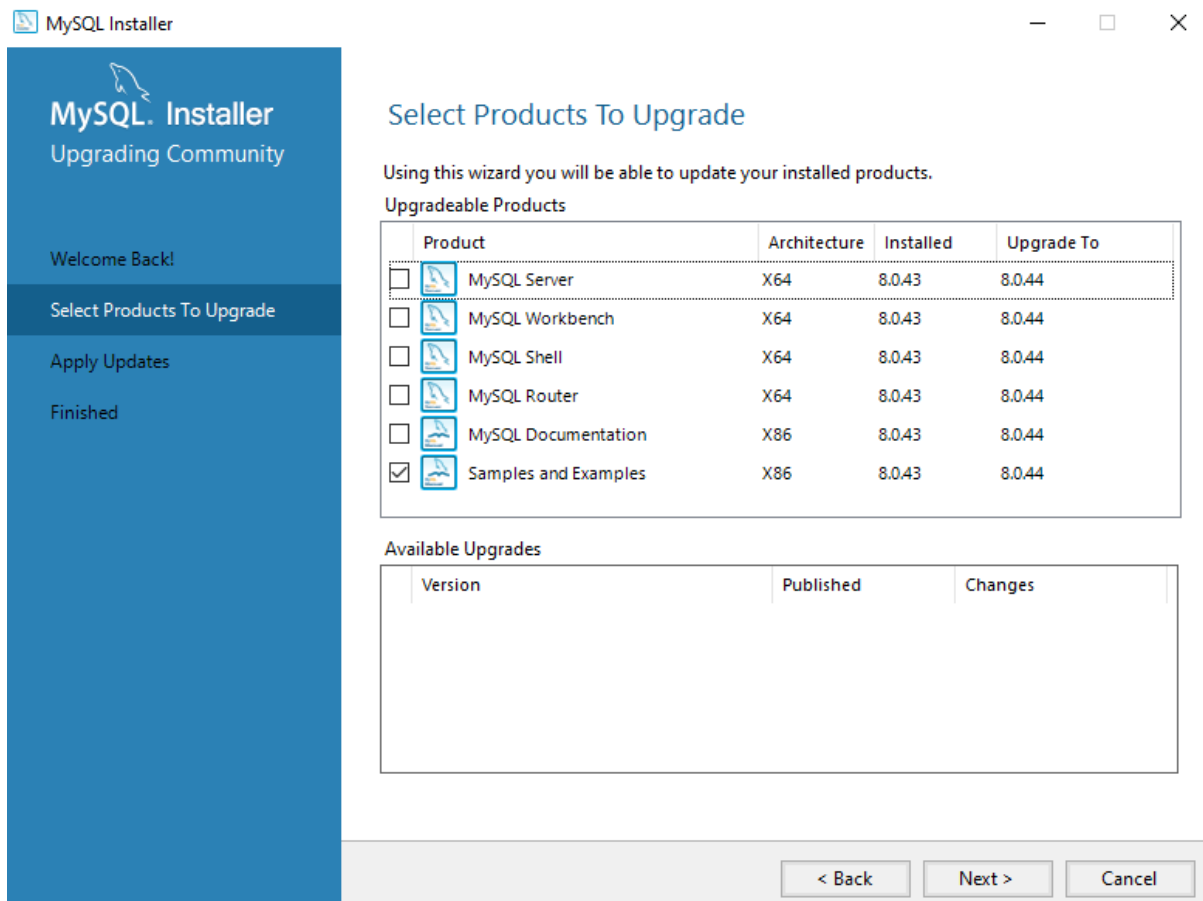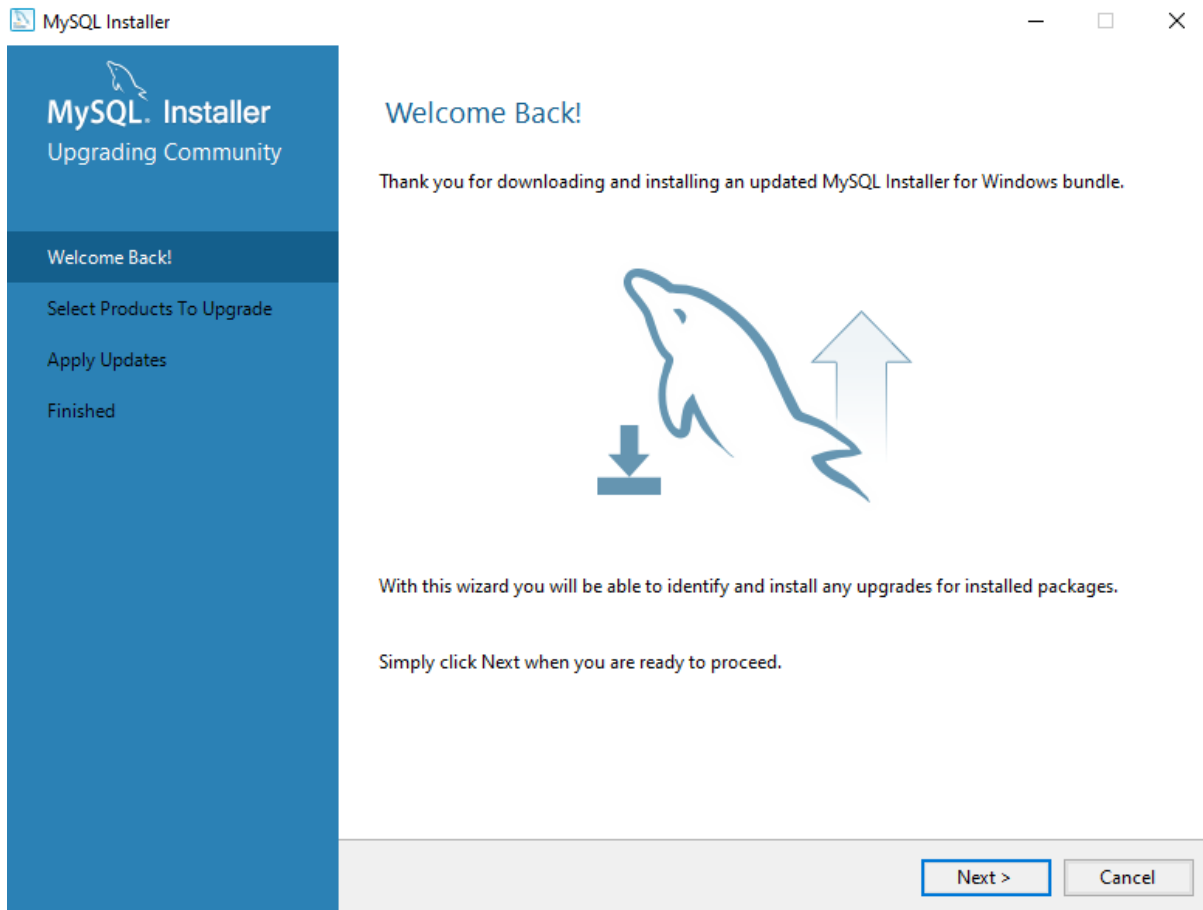
```
        \connect root@localhost

        See below
```


Figure 37: Connecting to root in mysql shell using \connect

MySQL Sample Databases

Download MySQL sample databases from here. MySQL Northwind database is here.

MySQL Installer

MySQL. Installer
Upgrading Community

Welcome Back!

Select Products To Upgrade

Apply Updates

Finished

Welcome Back!

Thank you for downloading and installing an updated MySQL Installer for Windows bundle.

With this wizard you will be able to identify and install any upgrades for installed packages.

Simply click Next when you are ready to proceed.

Next >   Cancel

MySQL Installer

MySQL. Installer
Upgrading Community

Welcome Back!

Select Products To Upgrade

Apply Updates

Finished

Select Products To Upgrade

Using this wizard you will be able to update your installed products.

Upgradeable Products

	Product	Architecture	Installed	Upgrade To
☐	MySQL Server	X64	8.0.43	8.0.44
☐	MySQL Workbench	X64	8.0.43	8.0.44
☐	MySQL Shell	X64	8.0.43	8.0.44
☐	MySQL Router	X64	8.0.43	8.0.44
☐	MySQL Documentation	X86	8.0.43	8.0.44
☑	Samples and Examples	X86	8.0.43	8.0.44

Available Upgrades

Version	Published	Changes

< Back   Next >   Cancel

################# Done ####################