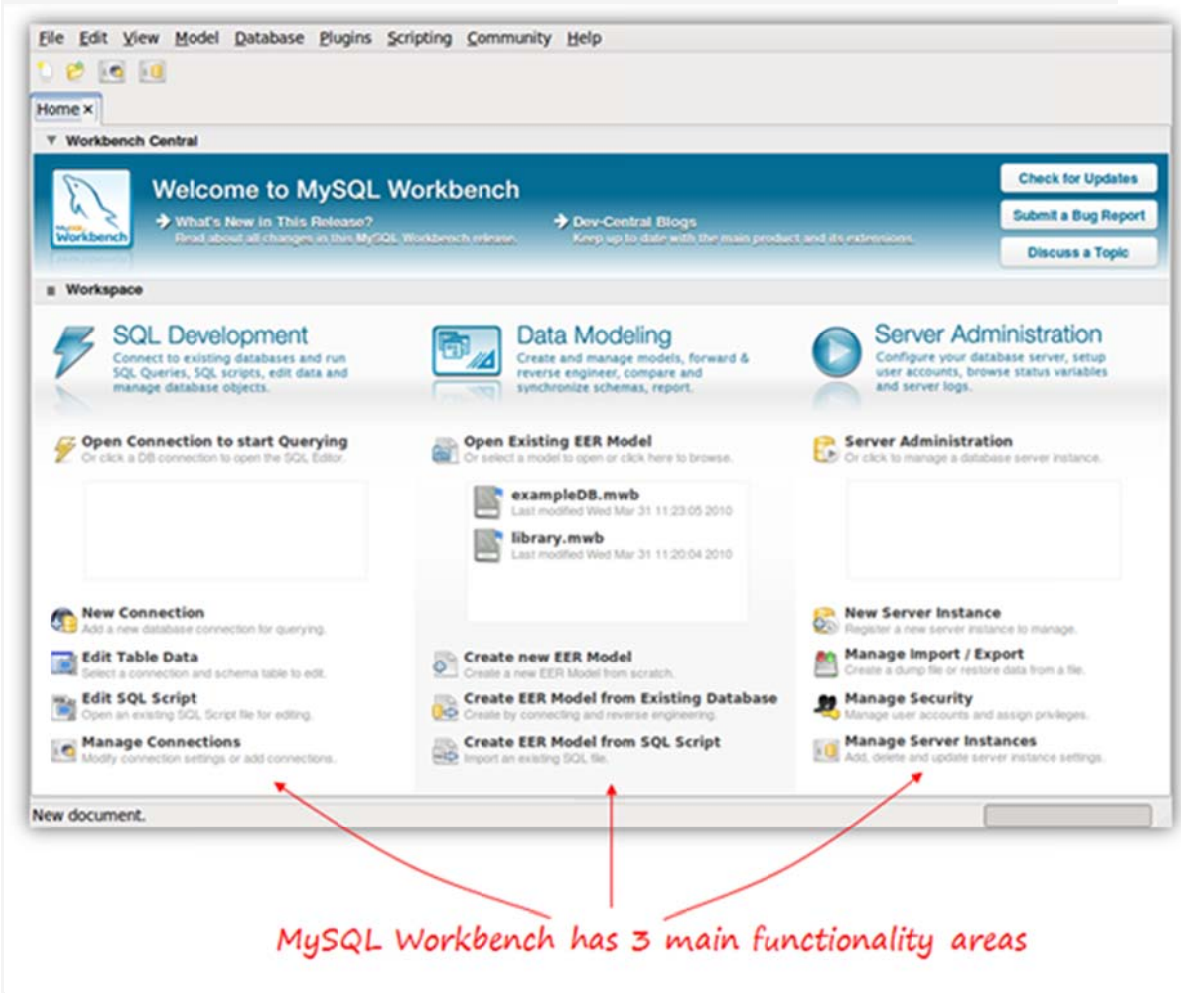
在今天的辅助教程里，你将会学到怎样用一个可视化数据库建模实用工具设计一个数据库图表和自动生成 SQL 语句。 特别说一下，我们将会回顾一下怎样用 MySQLWorkbench，一个交叉平台，可视化数据库设计工具。

什么是 MySQL Workbench?

MySQL Workbench 是由 MySQL 开发的强大的工具， 它有以下三个基本功能区域。

- **SQL Development**: 代替了 MySQL query browser（MySQL 查询浏览器）. 允许用户连接到现有得的数据库，编辑和执行 SQL 查询。
- **Data Modeling**: 完整的可视化数据库设计和建模
- **Database Administration**: 代替了 MySQL administrator. 图形界面启动和关闭服务，创建用户账户，编辑配置文件，等等。



在这个教程中，我们将会在 *Data Modeling* 上从头开始创建一个数据库，然后快速的看一下在 SQL 编辑器上执行 SQL 脚本并且在 MySQL 中创建一个数据库。

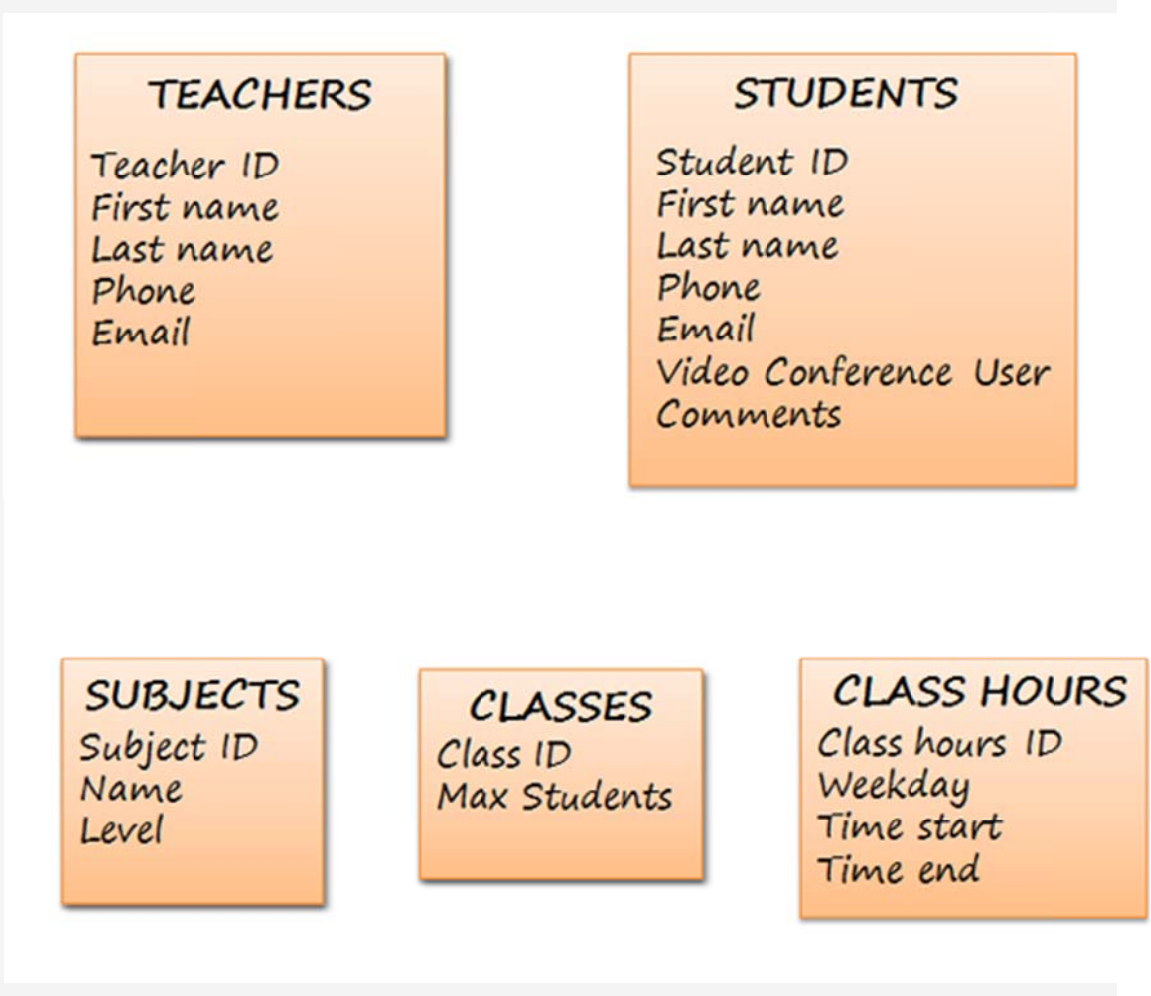
MySQL Workbench 可适用于 Windows，Linux 和 Mac OSX。 有两个不同的版本：*Community OSS Edition* （社区 OSS 版本）和 Commercial *Standard Edition*（商业标准

版本）。 社区版本是开源和 GPL 授权的，正如你所期望的。它功能齐全并且是我们在这篇文章里用到的。商业版本增加了一些其他的功能，像视图和模型验证或者文件生成。

备注：这个辅助教程是基于 Community OSS Edition 5.2 版本的(5.2.16)，在测试版本发行时写的(2010.4)。

开始创建数据库

为了学会怎样使用 MySQL Workbench，我们将会使用一个非常简单的数据库：在线课程作为案例。假如一组老师想给几门学科提供在线课程，使用 Skype 或者其他视频会议软件。对于我们这个小的项目，我们应该存储以下信息：



当设计我们的图表时，我们需要知道以下几组数据之间的关系，所以我们现在就应该想一下。

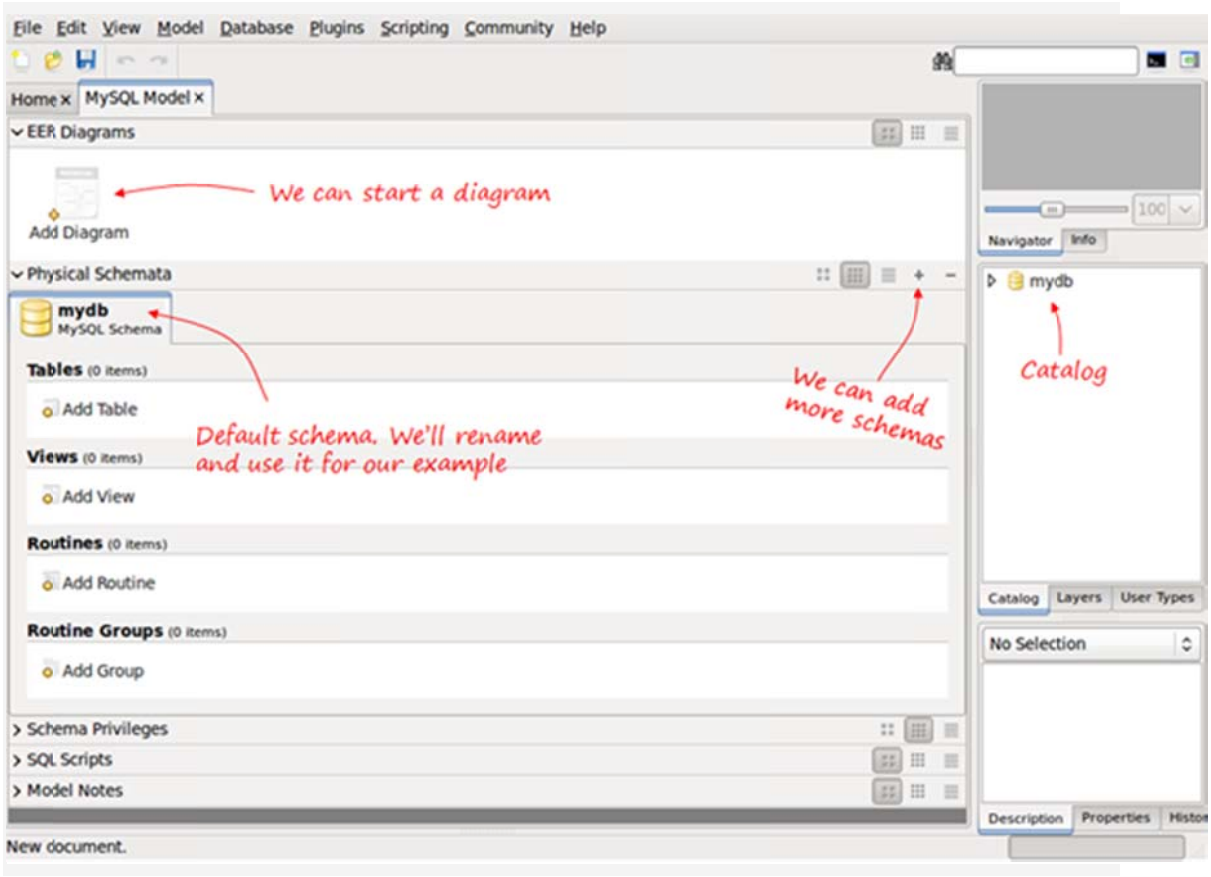
- 一个老师可以教很多学科

- 一个学科可以被很多老师教
- 每一个课程只能有一个老师
- 一个老师可以教很多班
- 一个同学可以参加几个班
- 一个班可以有很多学生
- 一个班可能有几个小时的课(一周内)
- 在某一特殊的天或小时中，可能同时有几个在线的班
- 一个班只关于一个学科
- 一个学科可能会在几个课程中被教

基于这一点，在下面的展示中我们就有了所有的信息。

把数据传到MySQL Workbench中

现在到了开始说Workbench的时候了。在主（Home）屏幕的数据模型部分，我们点击*‘Create new EER Model’*，下面的屏幕将会出现：

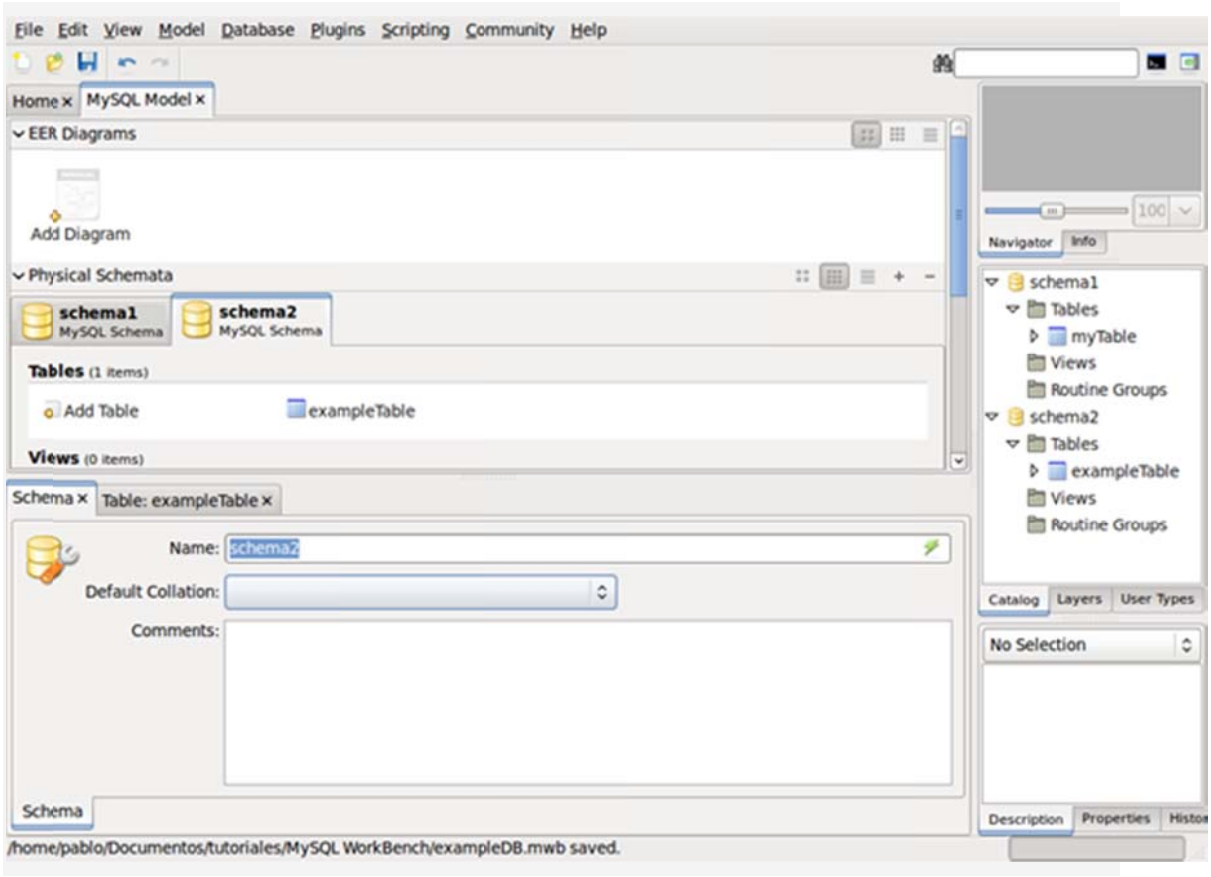


当我们新建一个数据库模型时，它包括了默认的 mydb 模式， 我们可以重命名它并且作为我们的数据库模式。一个数据库模型可以有几个不同的模式。右边的目录将会展示我们的模式的每一个元素，并且允许我们在需要的情况下拖拽这些元素。对于物理模式和 EER 图表有几个独立的部分，我们可能会混淆包含在一个数据库中的几个模式。接下来的部分将会解释这些概念和它们之间是怎样关联的。

区分概念

物理模式包含了所有需要定义数据库的部分：表、列、类型、索引、 约束， 等等。这是我们真正要定义的。每一个被添加在图表模型的对象都会在物理模式中显示 。也就是每一个图形化方式都会被定义在模式中。

在同一个数据库模型中我们可有几个模式，同样的，在一个 MySQL 服务器中也可以由几个数据库。每一个模式将会是一个 MySQL 数据库。例如，在下面的屏幕里我们有两个模式选项卡



如果我们要生成 SQL 脚本，我们将有两个独立的创建数据库声明—实际上创建视图和创建数据库是两个同义词：

```
view plaincopy to clipboard print?
1. CREATE SCHEMA IF NOT EXISTS `schema1`;
2. CREATE SCHEMA IF NOT EXISTS `schema2`;
```

“EER 代表了扩展的实体关系，EER 图表只是做模型数据和数据关系用的标准符号。

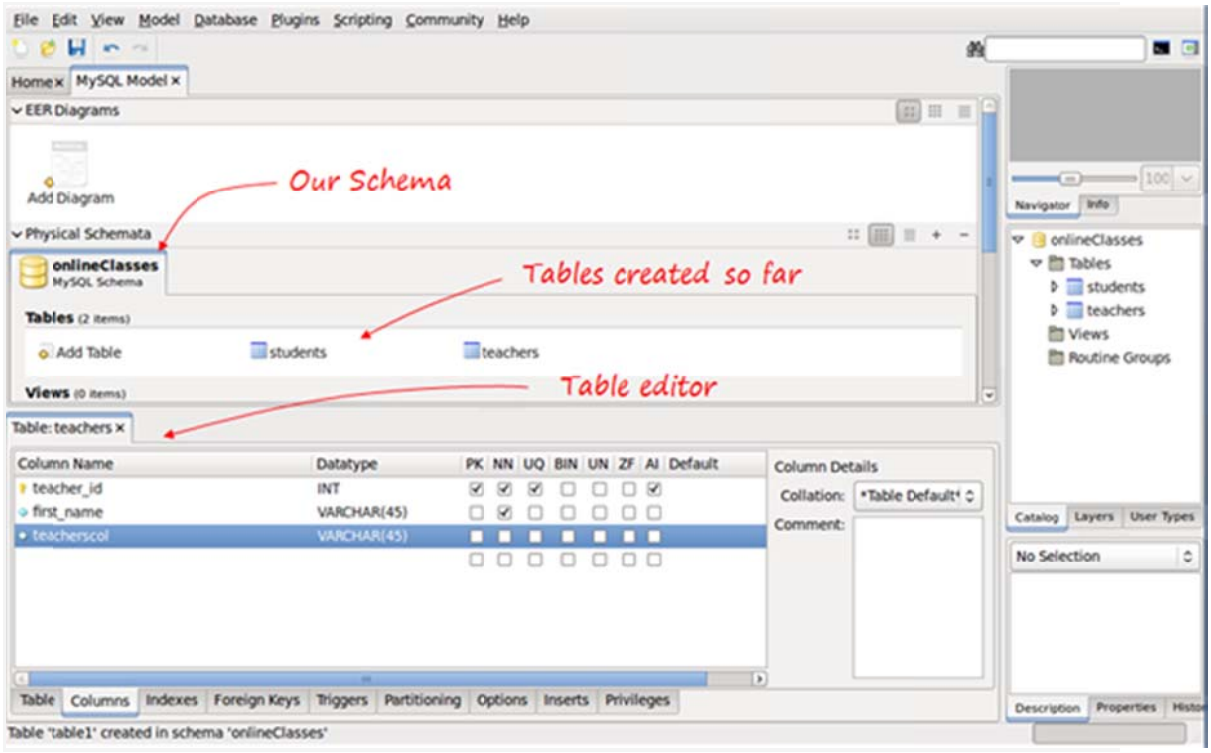
当用到 SHOW DATABASES 时，它们将会在 MySQl 服务器上作为数据库被列出来。

现在看一下什么是 EER 图表？EER 代表了扩展的实体关系，EER 图表只是做模型数据和数据关系用的标准符号。EER 模型可以很复杂，但是 MySQL Workbench 只用了所有可能图形化元素的一个子集，因为这个图表（在这个工具里）的目的是拥有每一个被映射到物理模式的元素。

我们可以用 EER 图表定义整个数据库，或者是其中的一小部分。例如，我们可以在一个模式中定义五个数据表然后新建一个图表（drigram）用可视化编辑工具去定义另外两个数据表。这个图表仅仅包含两个数据表，但是这两个数据表和其他的五个表都会被包含在模式中。

创建表

回到初始的实例，我们通过双击数据表的名字重命名它。在这一点上，有两个途径：我们可以用添加数据表图标向物理模式中添加数据表或者使用 EER 图表添加所有的数据表。我一般选择开始的时候添加 EER 图表并且创建可视化视图，但是为了展示怎样用两种方法创建数据表，我们将会用视图标签第一次创建两个数据库表然后接着用 EER 图表创建。当你点击 *Add Table* 图标，下面的数据表编辑界面会打开：



使用数据表编辑界面，我们改变数据表的名字，并切换到列标签（在下面的编辑器选项卡）进入我们的列。我们可以选择数据类型（这儿有一个下拉列表框关于所有的 MySQL 数据类型），分配默认的值，如果需要，我们有七个复选框去标记下面的任意一个内容。

- PK - Primary key（主键）
- NN - Not null（非空）
- UQ - Unique（唯一）
- BIN - Binary（二进制数）
- UN - Unsigned（无符号）

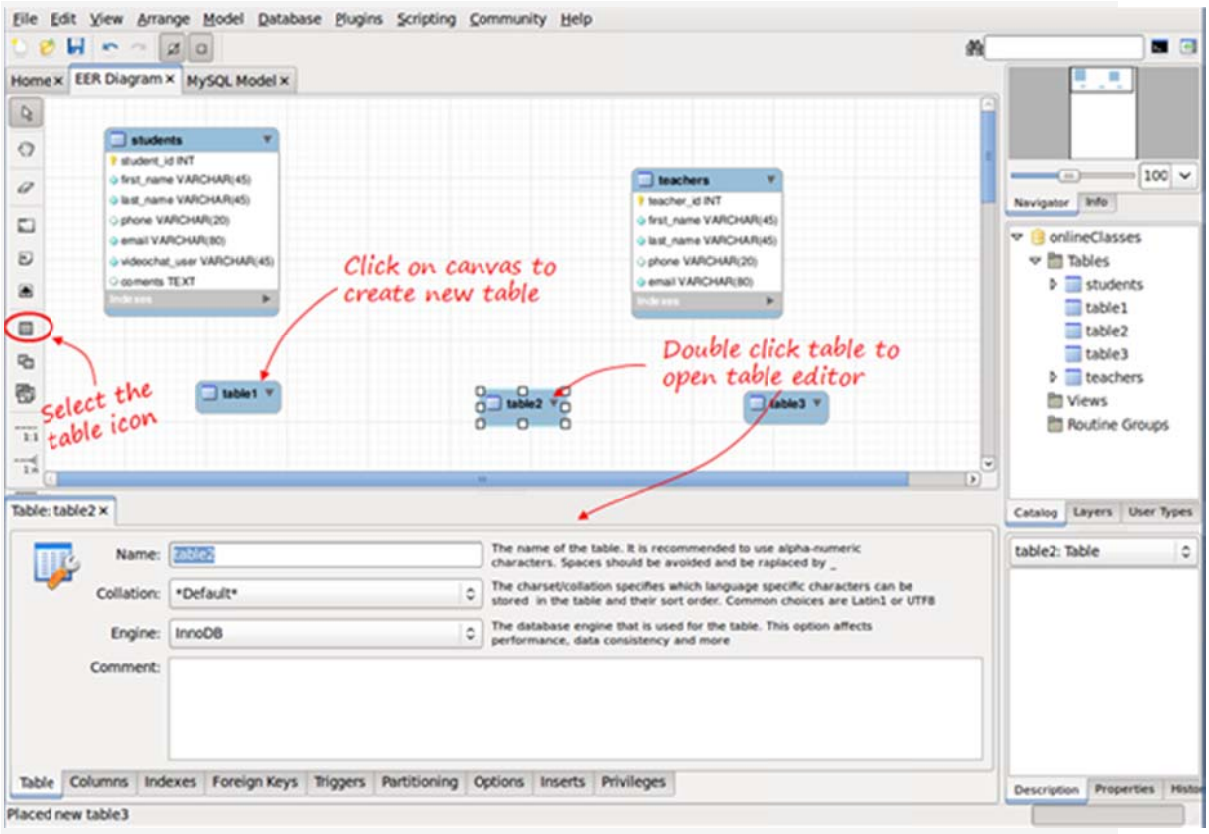
- ZF - Zero fill（补零）
- AI - Autoincrement（自动增量）

转到视图

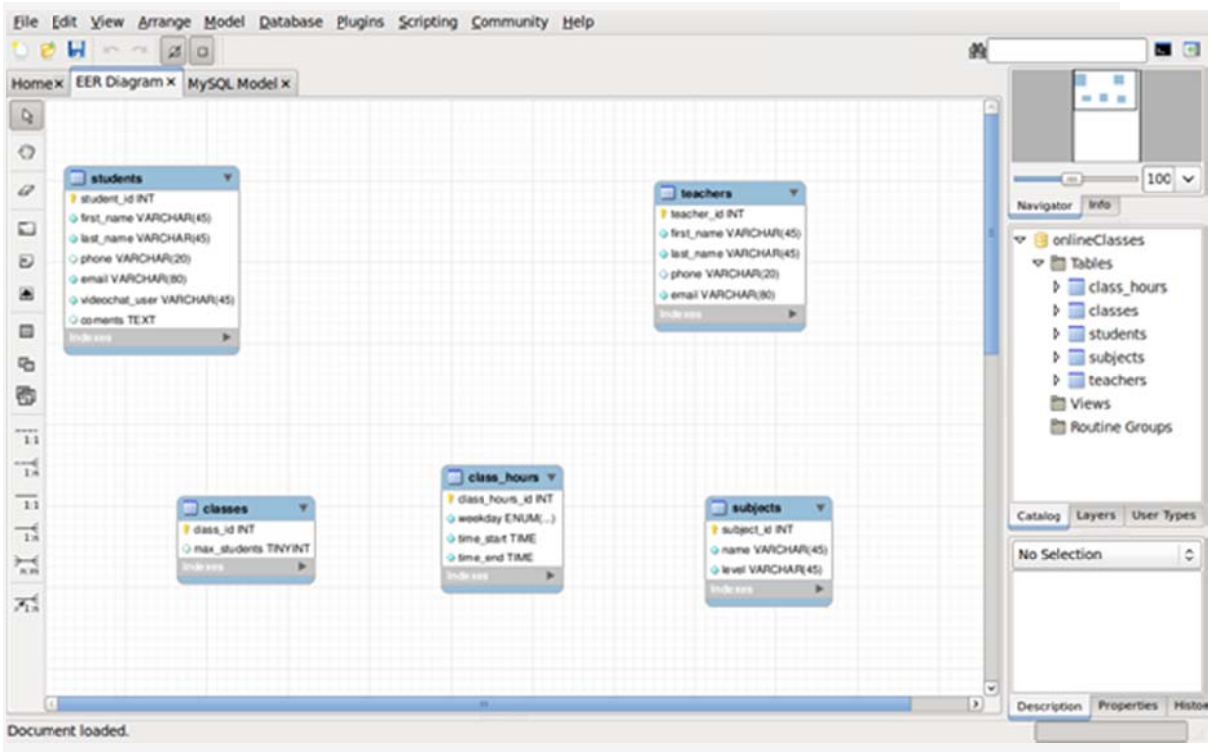
这是一种添加数据表的方式，当然我们也可以用 EER 图表创建它们。如果我们现在点击 *Add Diagram* 标记，我们将新建一个空的图表，这不是我们想要的，我们需要的是刚才在图表中建的两个数据表。

如果我们转到菜单处，选择 Model/Create Diagram from Catalog Objects,现在我们有了图表并且准备继续创建。

选择左边的数据表图标，指针指向一个小的数据表。下一步，单击容器的任何地方创建一个新的数据表。

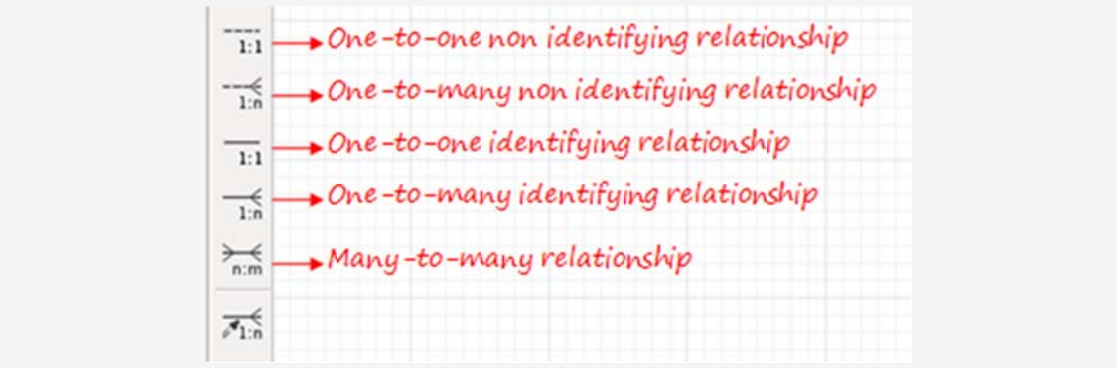


现在你只需要双击数据表，编辑器选项卡显示编辑的名称、列、类型等，像我们之前做的方式一样。向表中输入行的详细信息后，我们准备开始设计它们之间的关系。



Drawing Relationships

在左边的数值工具栏上，我们在创建表关系上有六个工具可供选择



不用担心最后一个，我们最后将会解释它。对于 1:1 和 1:n 关系，我们有两种类型的标记符号：定义和未定义。这是什么意思呢？

一种关系被认为是定义一个表完全依赖与另一个存在的表时，一个数据表的一行依赖于另一个数据表的一行。一个普遍的例子就是用一个分开的表存储用户的手机号。这可能在另一个表中是必须的，因为一个用户可能有几个手机号，但是手机号表的每一组是完全依赖用户表的，它是属于用户的。

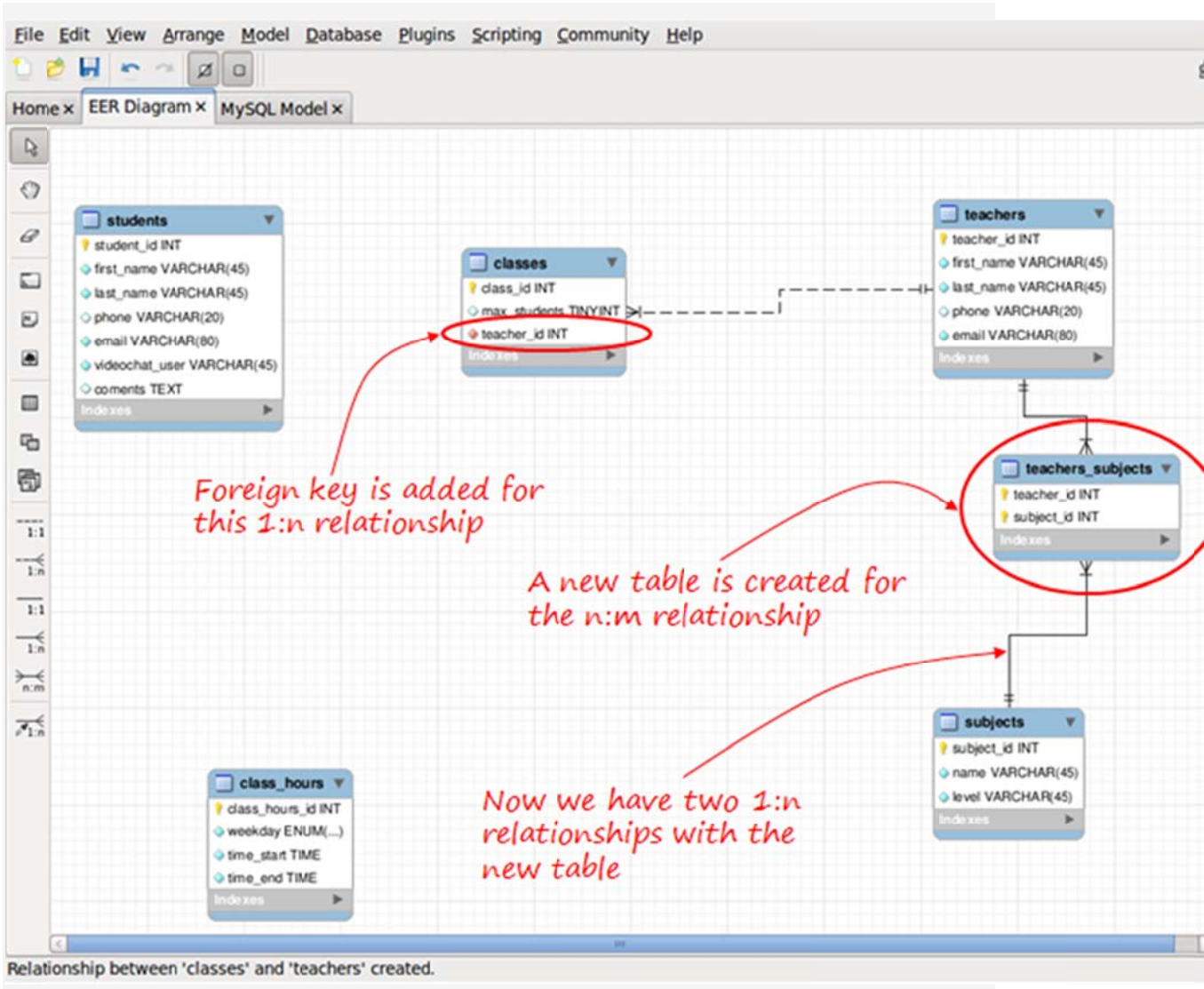
你可能意识到关系有一些蕴含式。如果我们想在 MySQL 中建立一个物理表，关系必须被某种方式映射。在表的关系映射中有几种规则。

1:1 关系:一个表的主键是另一个表的外键

1:n 关系:关系对应的表的主键是 n 关系对应的表的外键

n:m 关系:一个新表(关联表)被创建。主键是这两个原始表的主键的组成。

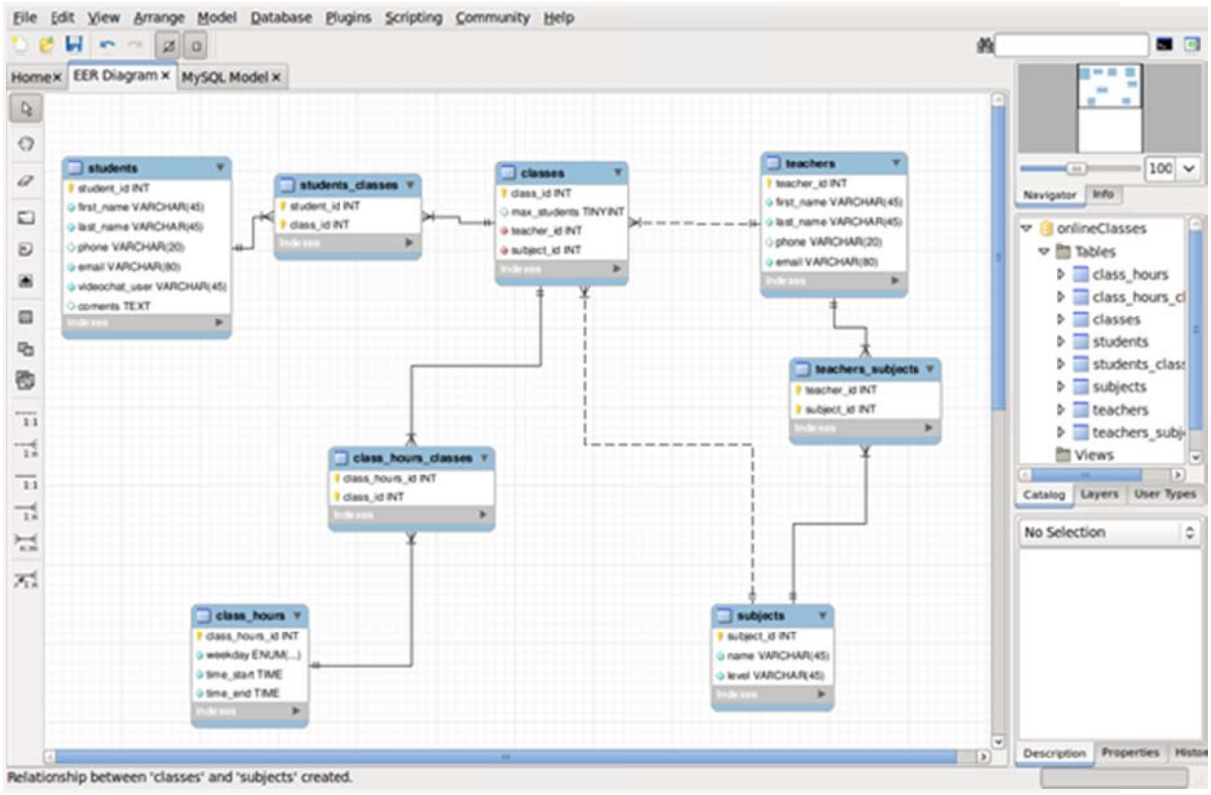
从 n:m 关系说明创建关联表时定义关系是比较典型的。这些数据表完全依赖于两个原始的表。而且_，在 1:1 和 1:n 的关系定义上，外键的引入将会成为那个表的主键一部分形成一个复合的主键。一个好的消息就是 MySQL Workbench 比我们更好的知道这些约束。我们只需要画线，外键和关联表将会被自动创建。我们也可以选择手动，一会将讲到。为了建立关系，单击图标，然后点击这两个表的关联。对于一对多关系，先点击多关系对应的表，_然后点击 1 关系对应的表。让我们看一下该怎样做对于 n:m 的 teachers-subjects 关系和 1:n 的 teachers-classes 的关系。



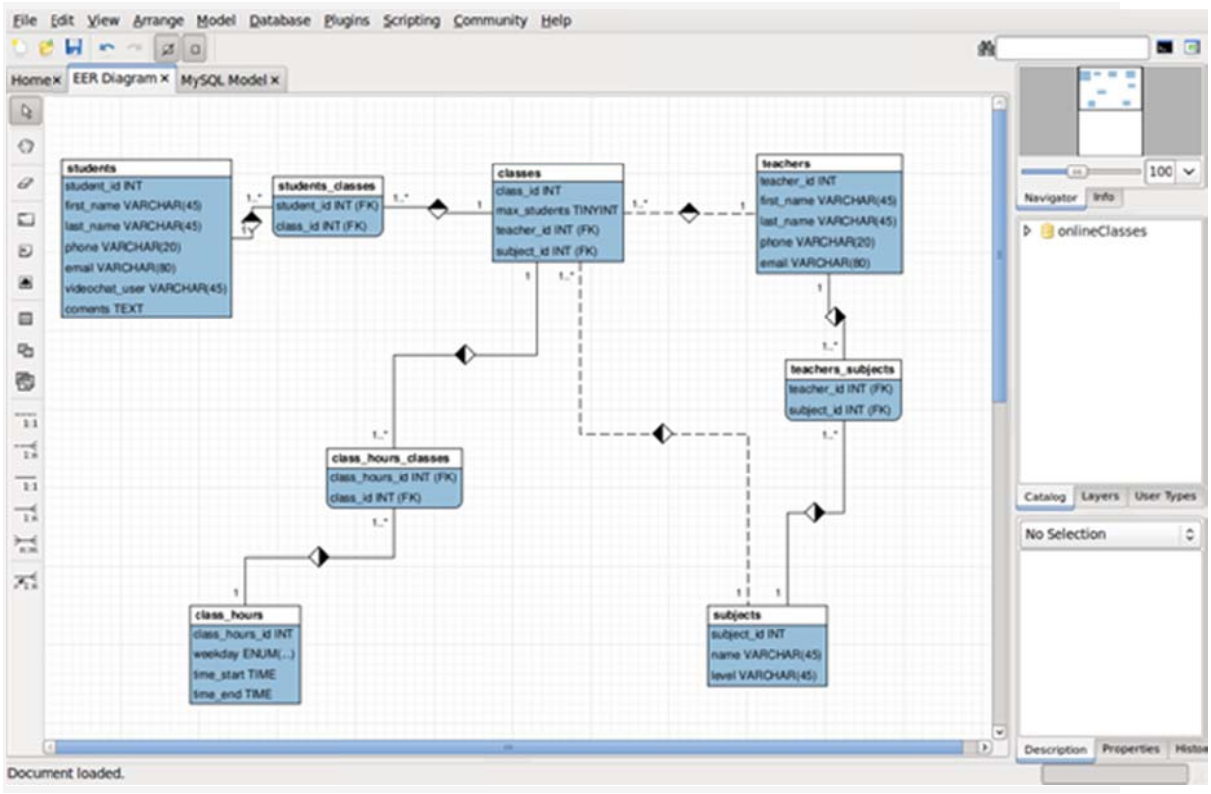
默认指定的外键名字和关联表可能会在 *Edit/Preferences/Model Tab* 上被全局改变或者只是在当前的项目 *Model/Model Options* 上。如果我们不想让表和外键被这种方式生成，我们可以用难以理解的第六个符号。

第六个符号用存在的列创建关系，意味着在你的表中已经包括了需要的外键并且建立了需要的关联表（n:m 映射表）。既然我们建立了这些关联表，就不需要 n:m 的关系了，1:n 就可以了。

如果我们把所有的关系都定义好了，我们的图表应该看起来像下面这样：



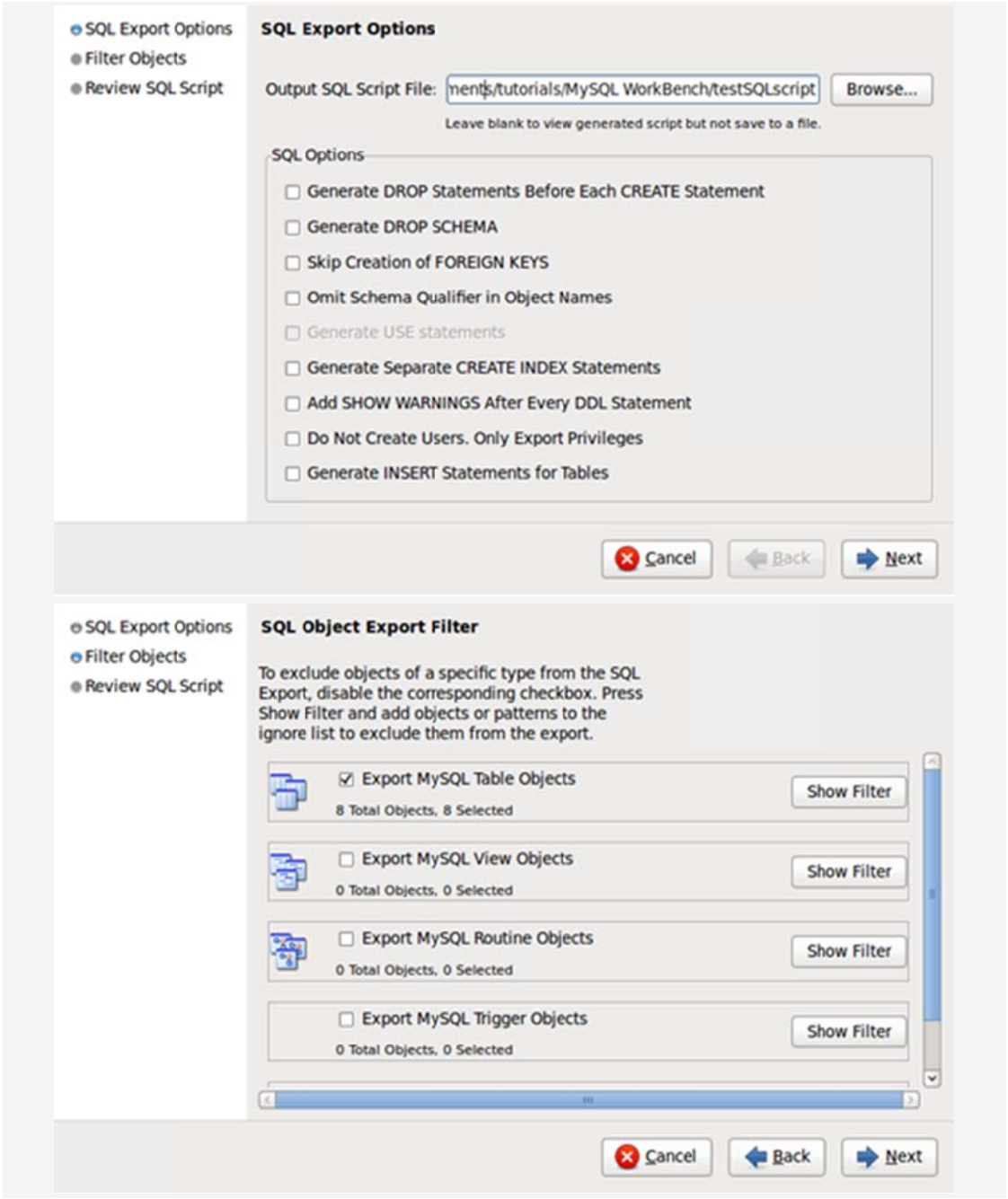
意识到我们一直在 MySQL Workbench 中使用默认的图表符号，但是你也可以在 *Model/Object Notation* 和 Model/Relationship Notation 改变它。这是我们在 Classic notation 模型中的案例:



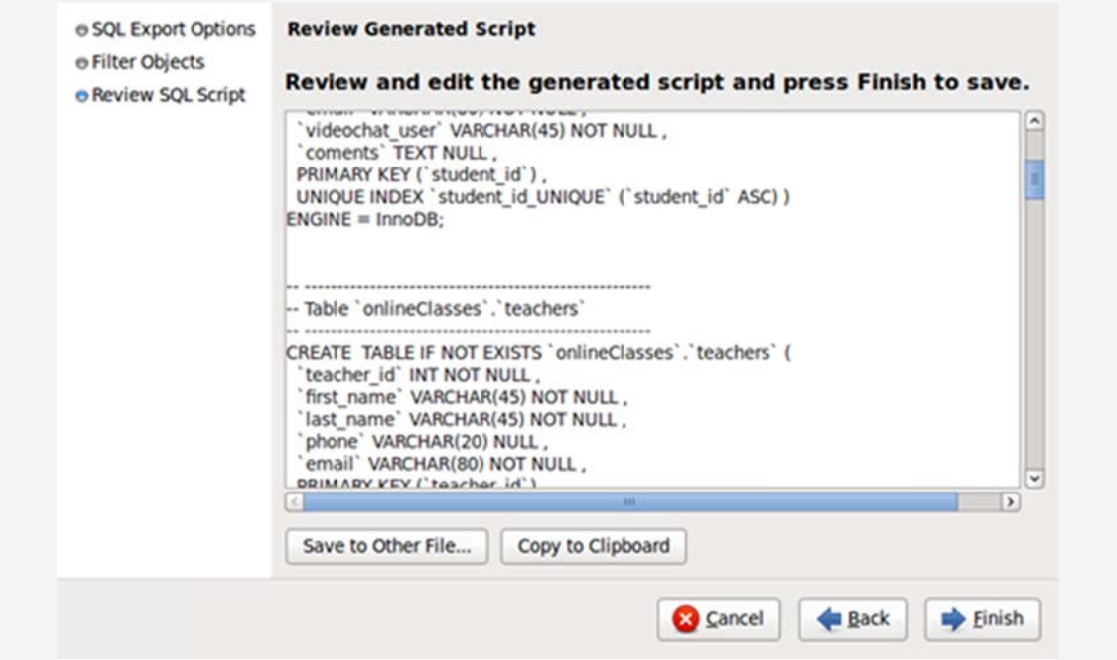
在这一点上，我们的模型是准备好的，我们可以生成 SQl 创建 MySQl 数据库。

生成SQL

选择 *File/Export/Forward Engineer SQL CREATE Script.* 在我们生成的文件的旁边只有三个向导屏幕：



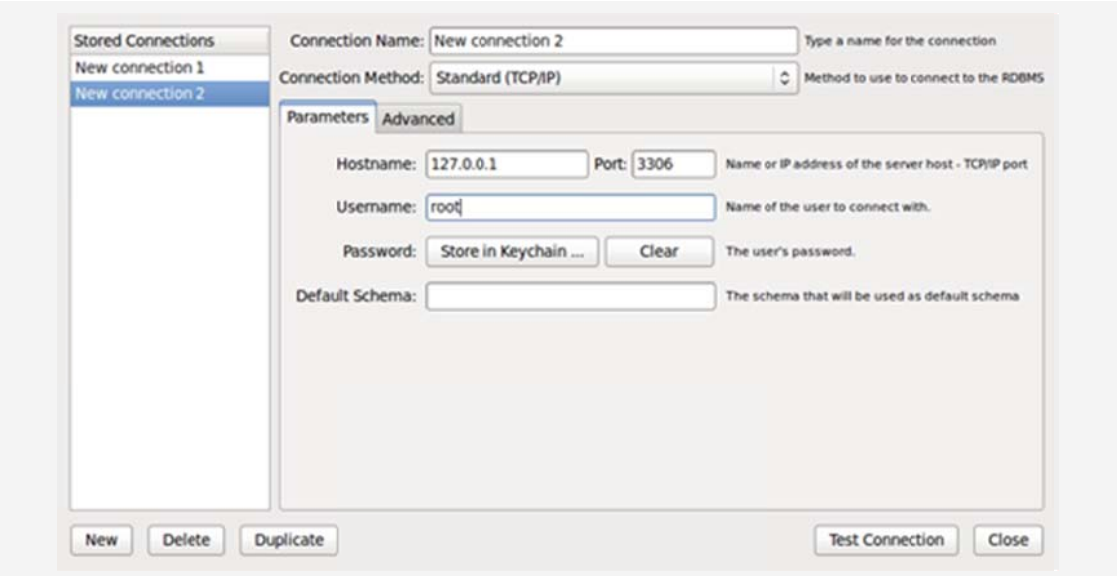
我们还可以在保存它之前选择复查和编辑生成的 SQL 语句。



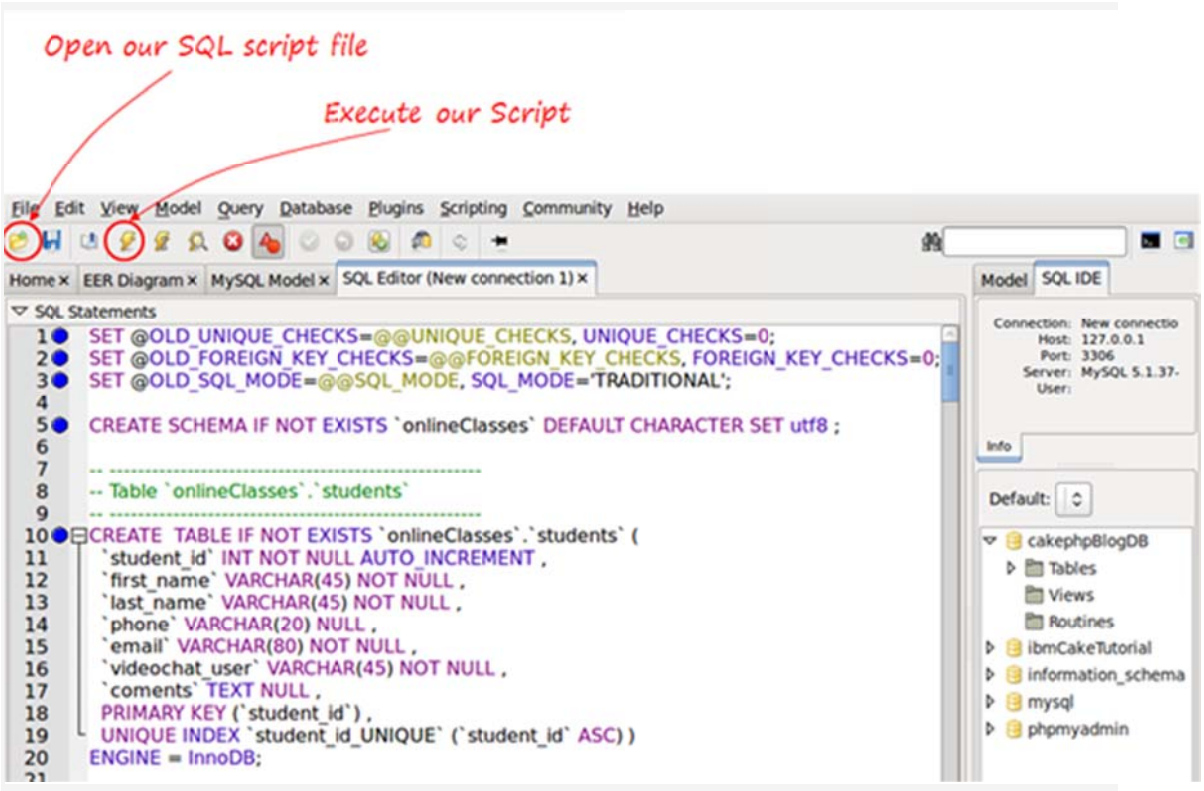
这样就可以了。点击 Finish，SQL 脚本将会被生成和保存。 现在我们可以用它做我们希望做的事情了。我们可以用基于命令行的 mysql client 载入它：mysql> SOURCE scriptName.sql，或者，我们可以用 MySQL Workbench 来完成这个工作，连接到我们的 MySQL 服务器然后执行脚本。

连到MySQL服务器

从菜单中选择 *Database/Manage Connections* ，点击 *NEW*.



如果你不想在这儿设置密码， 当需要的时候你会被提示。点击“Test Connection” 测试一下你的参数是否正确，然后点击 close。现在载入的脚本，我们可以用 SQL 编辑器进行编辑。在主菜单里选择 *Database/Query Database*; 会有窗体提示你选择一个连接，然后 SQL 编辑标签将会出现。



现在点击闪烁的图标执行 SQL 脚本，然后你的数据库将会被生成。

我们也可以直接从模型里选择生成 MySQl 数据库，不用涉及当前的文件在菜单中用 *Database/Forward Engineer*; 然而，我发现生成脚本并且在希望用到它的时候很用。

总结

MySQL Workbench 是一个给人印象很深的工具， 我们仅仅在数据模型部分看到一些基本的功能，并且只是在这个教程的第二个部分一瞥 SQL 编辑器。我们学会了怎样可视化的创建一个数据库，设计被作为文档保存的图表。你可以输出 PNG、SVg、 PDF 或者 PostScript 文件。谢谢阅读，让我知道你是怎样想的！