Database Concepts

8th Edition

David M. Kroenke • David J. Auer

Scott L. Vandenberg • Robert C. Yoder

Online Appendix C

Getting Started with MySQL 5.7 Community Server

VP Editorial Director: Andrew Gilfillan
Senior Portfolio Manager: Samantha Lewis
Content Development Team Lead: Laura Burgess
Program Monitor: Ann Pulido/SPi Global
Editorial Assistant: Madeline Houpt
Product Marketing Manager: Kaylee Carlson
Project Manager: Katrina Ostler/Cenveo® Publisher Services
Text Designer: Cenveo® Publisher Services
Interior design: Stock-Asso/Shutterstock; Faysal Shutterstock
Cover Designer: Brian Malloy/Cenveo® Publisher Services
Cover Art: Artwork by Donna R. Auer
Full-Service Project Management: Cenveo® Publisher Services
Composition: Cenveo® Publisher Services
Printer/Binder: Courier/Kendallville
Cover Printer: Lehigh-Phoenix Color/Hagerstown
Text Font: 10/12 Simoncini Garamond Std.

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on the appropriate page within text.

Library of Congress Cataloging-in-Publication Data

Kroenke, David M., 1948- author. | Auer, David J., author.
Database concepts / David M. Kroenke, David J. Auer, Western Washington University, Scott L. Vandenberg, Siena College, Robert C. Yoder, Siena College.
Eighth edition. | Hoboken, New Jersey : Pearson, [2017] | Includes index.
LCCN 2016048321| ISBN 013460153X | ISBN 9780134601533
LCSH: Database management. | Relational databases.
LCC QA76.9.D3 K736 2017 | DDC 005.74--dc23
LC record available at https://lccn.loc.gov/2016048321

Appendix C — 10 9 8 7 6 5 4 3 2 1

Appendix Objectives:

- Learn how to install MySQL 5.7 Community Server
- Learn how to install MySQL Workbench
- Learn how to install the MySQL ODBC/Connector
- Learn how to create a database in MySQL 5.7 Community Server Edition
- Learn how to submit SQL commands to create table structures
- Learn how to submit SQL commands to insert database data
- Learn how to submit SQL commands to query a database
- Learn how to import Microsoft Excel worksheet data into a database
- Learn how to use Microsoft Access as an application development platform
- Learn how to create database designs in the MySQL Workbench version 6.3.8

What Is the Purpose of This Appendix?

MySQL Community Server is an open source, freely downloadable, enterprise-class DBMS that has been around for many years. In November 2005, MySQL 5.0 was released, and, as of this writing, MySQL 5.7 is the generally available (GA) release. In February 2008, Sun Microsystems completed its acquisition of MySQL AB, the company that created and owned MySQL. Subsequently, Oracle Corporation acquired Sun Microsystems in a deal that was finalized on January 27, 2010, after European Commission approval on January 21, 2010 (see http://www.sun.com/third-party/global/oracle/). Thus, the Oracle Corporation now owns MySQL in addition to its flagship Oracle Database product.[1] MySQL, while not having as many

[1] For information about Oracle Database Express Edition 11*g* Release 2, see online Appendix B, "Getting Started with Oracle Database XE."

features as SQL Server, has become widely used and very popular as a DBMS supporting Web sites running the Apache Web server. The MySQL Community Server version and MySQL Workbench graphical user interface (GUI) utility are free.

Why Should I Learn to Use MySQL?

For the purposes of this book, the most important reason to learn to use MySQL is that MySQL really handles SQL well. All the SQL commands and keywords in Chapter 3 and Appendix E marked "Does Not Work with Microsoft Access ANSI-89 SQL" will work with MySQL. There will still be minor variations in some SQL statements, but this is typical of DBMS products.

What Will This Appendix Teach Me?

As its title implies, this appendix is designed to get you started creating databases and running SQL commands and **SQL scripts** (which are related groups of SQL commands) so that you can use a more robust SQL environment than provided by Microsoft Access.

What Won't This Appendix Teach Me?

The material in this appendix does not go beyond the information necessary to get you started using MySQL. There are many important MySQL topics not covered here, including stored procedures, triggers, backups and restores, and database security. Some of these topics are covered in David M. Kroenke and David J. Auer, *Database Processing: Fundamentals, Design, and Implementation*, 14th edition (Upper Saddle River, NJ: Prentice Hall, 2016).

How Do I Install MySQL 5.7 Community Server?

MySQL runs on Linux, UNIX, NetWare, and even the Microsoft Windows operating systems. The various versions of Microsoft SQL Server (including the free SQL Server 2016 Developer Edition discussed in Appendix A), like all other Microsoft products, run only on Microsoft operating systems. If you are not using a Microsoft operating system, MySQL is an obvious contender for your DBMS of choice. MySQL is intended for general use and can be downloaded for free. You need to download at least the first two of the following three programs listed on the next page, but we recommend that you download all three.

IMPORTANT NOTE: If you are using a **Windows operating system**, you should ***only*** install these programs by using the **MySQL Installer for Windows**, which can be downloaded from http://dev.mysql.com/downloads/installer. Do *not* try to install the programs separately on a Windows OS computer! If you do so, you may find that some functionality or connectivity between the programs is missing!

1. **MySQL Community Server Edition**.[2] Download the most current generally available version for your operating system. As this appendix was written, MySQL 5.7 Community Server was the most recent version generally available. Therefore, we will base this appendix on that version.
2. **MySQL Workbench**.[3] The MySQL Workbench has replaced the MySQL GUI Tools (the MySQL Query Browser and MySQL Administrator) as the graphical administration and SQL command utility for MySQL.[4] In addition, it can be used for creating database designs as described in Chapter 5.
3. **MySQL Connector/ODBC**. This provides the ODBC programs necessary to provide Web application connectivity to a MySQL database as described in Chapter 7.
4. **MySQL for Excel**. This provides the Microsoft Excel component needed to export data in Microsoft Excel into a table in a MySQL database, as described later in this appendix.

By The Way

If you look at the various Web sites for MySQL, you will find that the name of the **MySQL Community Server** edition varies from place to place. On the main Oracle MySQL Web site (see http://www.mysql.com/products), it is referred to as *MySQL Community Edition*. On the MySQL Web site that we use to download the MySQL products (see http://dev.mysql.com/downloads/mysql), it is referred to as both *MySQL Community Edition* and *MySQL Community Server*. On the MySQL Web site that we use to download the MySQL Installer for Windows (see http://dev.mysql.com/downloads/installer), it is referred to as just the *MySQL Server*.

So, what shall we call it? In *Database Concepts* (8th edition) and all the accompanying online appendices, we use the term **MySQL Community Server**.

---

[2] This appendix, which was finalized after *Database Concepts* (8th edition) itself went to press, uses MySQL 5.7 Community Server version 5.7.17. The screenshots in *Database Concepts* itself are based on MySQL 5.7 Community Server version 5.7.13. You may notice some slight variations in the screenshots from the two versions.

[3] This appendix, which was finalized after *Database Concepts* (8th edition) itself went to press, uses MySQL Workbench 6.3.7 CE. The screenshots in *Database Concepts* Chapter 5 are based on MySQL Workbench 6.3.6 CE. You may notice some slight variations in the screenshots from the two versions.

[4] The MySQL Server GUI Tools reached "end of life" (EOL) on December 18, 2009. See the announcement at http://dev.mysql.com/support/eol-notice.html. These tools are still available at http://downloads.mysql.com/archives/gui/.

MySQL and its associated utilities are very easy to install. At this point, install and configure MySQL 5.7 Community Server and the MySQL workbench by (1) using the MySQL Installer for Windows if you are using a Microsoft Windows OS or (2) downloading and installing the separate programs for non-Windows OSs. Install MySQL 5.7 Community Server as a *Developer Machine* when asked during the installation process. MySQL Workbench will also be installed as part of the process. We will discuss the MySQL Connector/ODBC later in this appendix.

Note that during the installation and configuration of MySQL Community Server, you will be asked to provide a password for the *root* user account. **Root** is the name of the default MySQL administrator account. Be sure you provide the password when prompted, and be sure to remember this password! The username *Root* comes from the UNIX and Linux operating systems, where it is the name of the default system administrator account. Be aware that MySQL 5.7 is an enterprise-class DBMS, and it is therefore much more complex than Microsoft Access. Further, it does not include application development tools, such as form and report generators.

If you are using a Microsoft Windows operating system, we recommend that you download the **MySQL Installer for Windows**, which packages current versions of MySQL 5.7 Community Server, the MySQL Workbench, several MySQL connectors (including the ODBC connector needed for the Web application work in Chapter 7), other utilities, samples, examples, and documentation together with an installation utility that controls which products are actually installed. The MySQL Installer for Windows can be downloaded from http://dev.mysql.com/downloads/. There is a separate version of the MySQL Installer for Windows for each major version of MySQL, and the version we are using here is for MySQL 5.7. Before running the MySQL Installer for Window, you need to:

1. Install the **.Net Framework 3.5**. This can be done in **Control Panel**. Select **Programs | Programs and Features | Turn Windows features on or off** to launch the **Add Roles and Features Wizard** to install .NET Framework 3.5 in Features (interestingly, .NET Framework 4.5 is installed by default, but .NET Framework 3.5 is not).

2. Download and install the **Microsoft Visual C++ Redistributable Package for Visual Studio 2013** (32-bit or 64-bit version depending upon your operating system) from http://www.microsoft.com/en-us/download/details.aspx?id=40784. Windows 10 may already come with the 2013 Redistributable Package.

3. Download and install the **Microsoft Visual Studio 2010 Tools for Office** (32-bit or 64-bit version depending upon your operating system) from https://www.microsoft.com/en-us/download/details.aspx?id=48217. This is needed for the MySQL for Excel utility.

4. **IMPORTANT!** Before installing MySQL, you will need to know whether your version of Microsoft Access is 32-bit or 64-bit, so that we can install the correct version of the ODBC Connector drivers. To do this:

a. Start Microsoft Access 2016 and double-click the **Blank desktop database** template icon. Click the **File** command tab, and the **Account** to get the **Office Product Information Screen**. Then click the **About Access** button.

b. The **About Microsoft Access 2016** dialog box is displayed, and the version (32-bit or 64-bit) appears at the end of the very top line. Make a note of which version it is, and then click the **OK** button.

c. Close Microsoft Access 2016.

d. Note that when we double-clicked the Blank desktop database icon creates a Microsoft Access 2016 database named Database1.accdb in the **This PC | Documents** folder. We will not use this database, so open File Explorer and delete it. Close File Explorer when you have done so.

Installing MySQL Community Server 5.7

1. To start the actual installation process, run the downloaded *mysql-installer-community-5.7.17.0.msi* file from your browser window, or enter open File Explorer and double-click the file in the Downloads folder. If a Security Warning dialog box is displayed, click the **Run** button. When the **User Account Control** dialog box is displayed, click the **Yes** button.

2. The **MySQL Installer** dialog box opens, and the *License Agreement* screen is displayed, as shown in Figure C-1(a). Read the agreement, check the **I accept the license terms** checkbox, and then click the **Next** button.

3. The *Choosing a Setup Type* screen is displayed, as shown in Figure C-1(b). Since we want to install the minimum set of MySQL components, select the **Custom** radio button, and click the **Next** button.

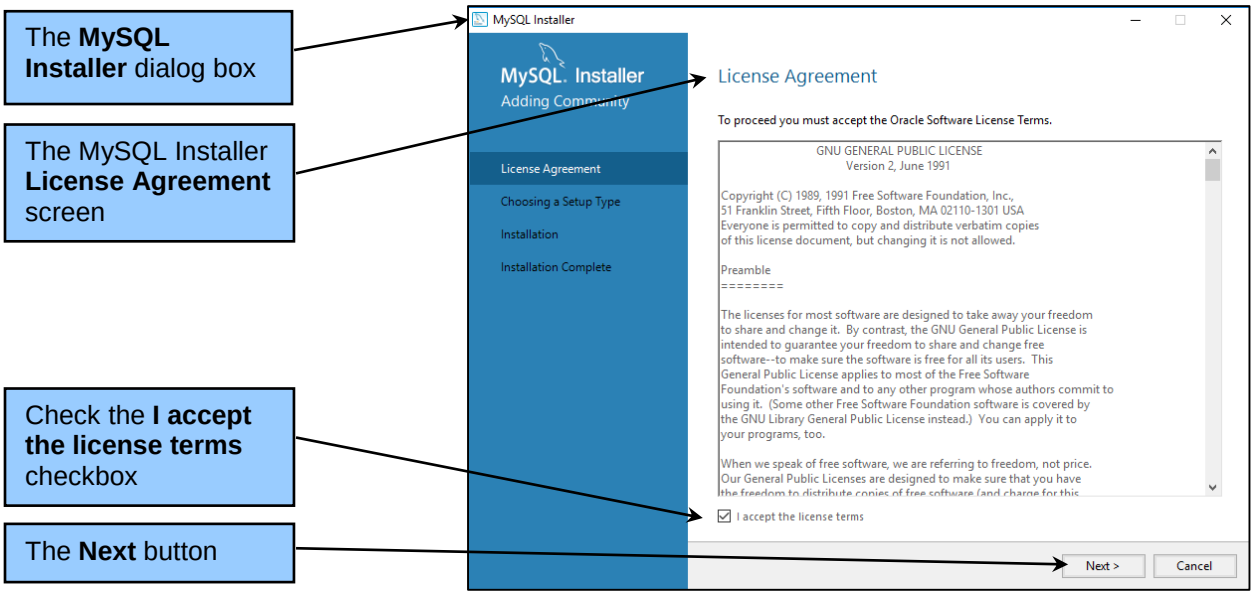


Figure C-1(a) — The MySQL Installer License Agreement Screen

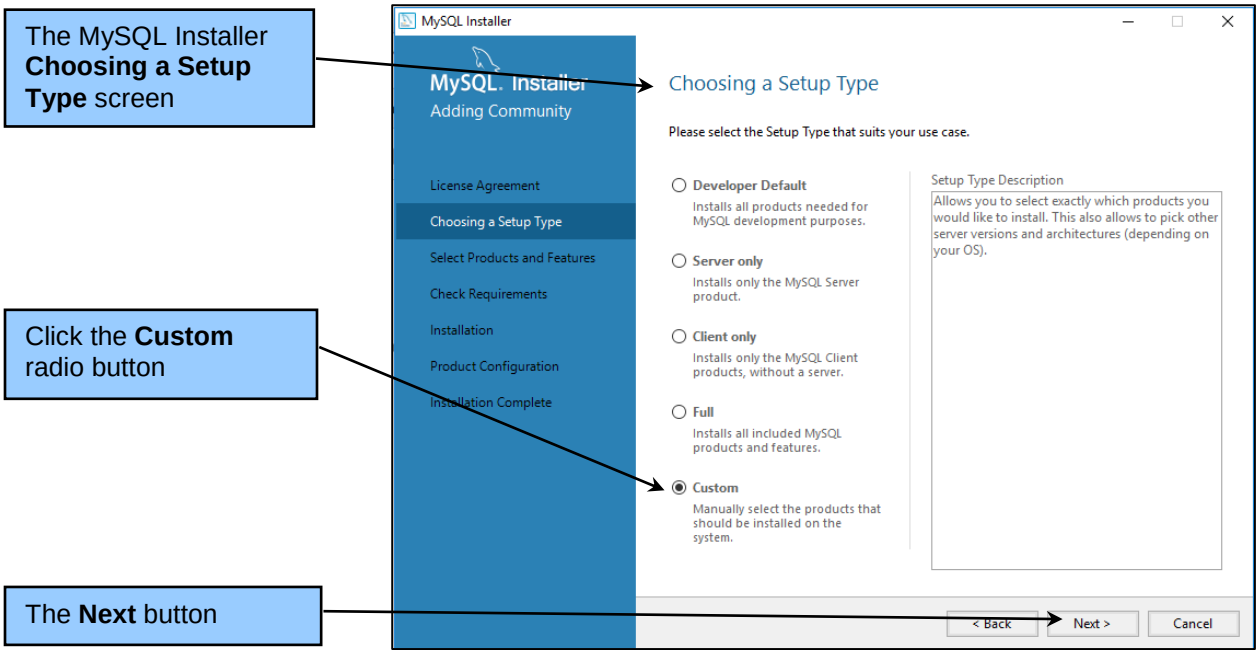


Figure C-1(b) — The MySQL Installer Choosing a Setup Type Screen

4. The *Select Products and Features* screen is displayed, as shown in Figure C-1(c). The first selection will be MySQL Community Server 5.7 itself. Expand the **MySQL Servers** option as shown in the figure. Select the version of **MySQL Server 5.7** that matches your operating system (we are using a 64-bit version of Windows 10, so we have chosen the X64 version—tTo determine which one you are using, use Settings | System | About), and then click the **right-facing arrow** button to add MySQL Server 5.7 to the *Products/Features To Be Installed* list.

5. Staying on the *Select Products and Features* screen, expand the **Applications** options as shown in Figure C-1(d). Select the version of **MySQL Workbench 6.3** that matches your operating system (we are using a 64-bit version of Windows 10, so we have chosen the X64 version), and then click the **right facing arrow** button to add MySQL Workbench to the *Products/Features To Be Installed* list. Although we will not install it, the **MySQL Notifier** utility is useful, and you may want to install it on your computer. It is available as an Applications option. We will also install **MySQL for Excel**, so add it as well. We will use it later in this appendix to import Microsoft Excel data into a MySQL database.

6. Staying on the *Select Products and Features* screen, expand the MySQL Connectors options as shown in Figure C-1(e). Based on whether you have a 32-bit or 64-bit version of Microsoft Access 2016 (as you determined earlier in this appendix), select the matching version of **Connector/ODBC 5.3**, and then click the **right-facing arrow** button to add Connector/ODBC 5.3 to the *Products/Features To Be Installed* list. Select 32-bit ODBC drivers if your version of

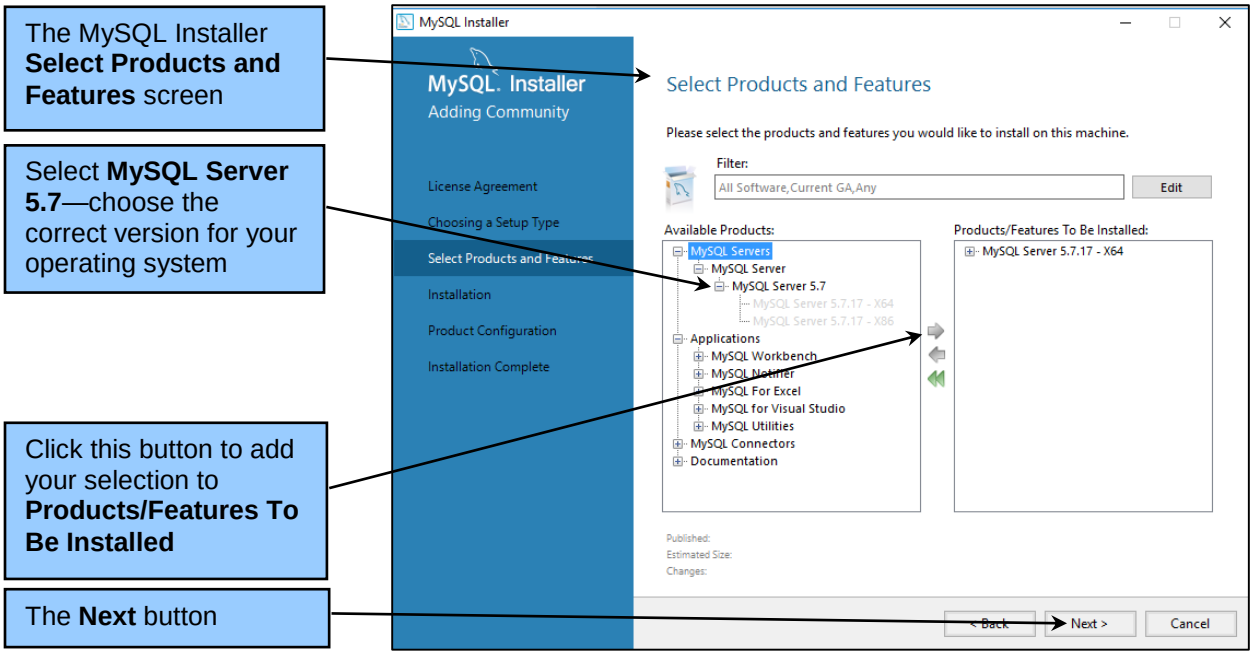


Figure C-1(c) — The MySQL Installer Select Products and Features Screen

Microsoft Access is 32-bit, 64-bit drivers if your Access version is 64-bit. Installing the correct version will avoid having driver mismatch problems between Windows 10 (64-bit) and Microsoft Access (can be 32-can or 64-bit). **REMINDER:** If you are using a **Windows operating system**, you should ***only*** install the MySQL Connector/ODBC (and any of the other MySQL Connectors available from MySQL if you need them) by using the **MySQL Installer for Windows.**

7. Staying on the *Select Products and Features* screen, expand the **Documentation** Options as shown in Figure C-1(f). Select the **MySQL Documentation 5.7** option then the 5.7.17 sub-option (at this time there is only a 32-bit version available—it will also run on a 64-bit operating system), and then click the **right-facing arrow** button to add MySQL Documentation to the *Products/Features To Be Installed* list.

8. Click the **Next** button. The MySQL Installer checks your computer to make sure all installation requirements are met. If any requirement test fails, a message is displayed. If you have not installed the **Microsoft Visual C++ Redistributable Package for Visual Studio 2013** or the **Microsoft Visual Studio 2010 Tools for Office** as discussed above, you will get an error message at this point. If all requirements are met, the MySQL Installer *Installation* screen is displayed, as shown in Figure C-1(g). This screen summarizes all the products that will be installed.

9. Click the **Execute** button. The MySQL Installer installs all the selected products, as shown in the Figure C-1(h). The installation status for each product is marked as Complete after it is installed.

10. After the product installation for all products is complete, click the **Next** button. The *Product Configuration* screen is displayed, as shown in Figure C-1(i). We will now configure our installation of MySQL Server 5.7.

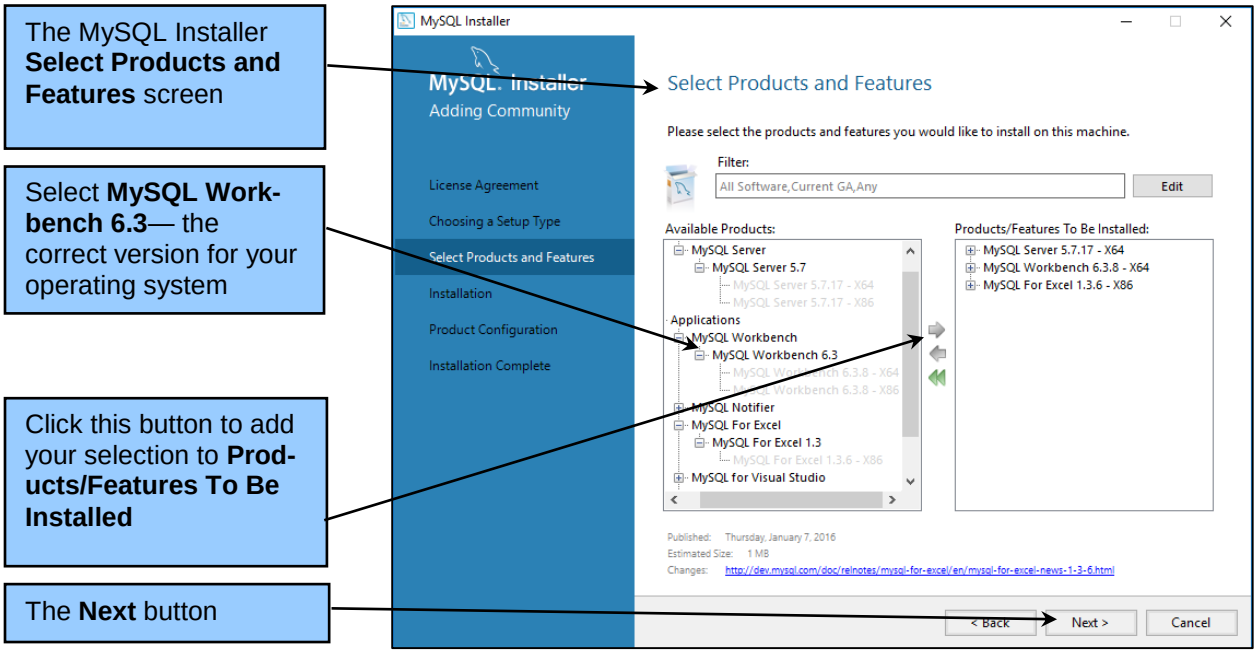


Figure C-1(d) — The MySQL Installer Select Products and Features Screen with Expanded Applications

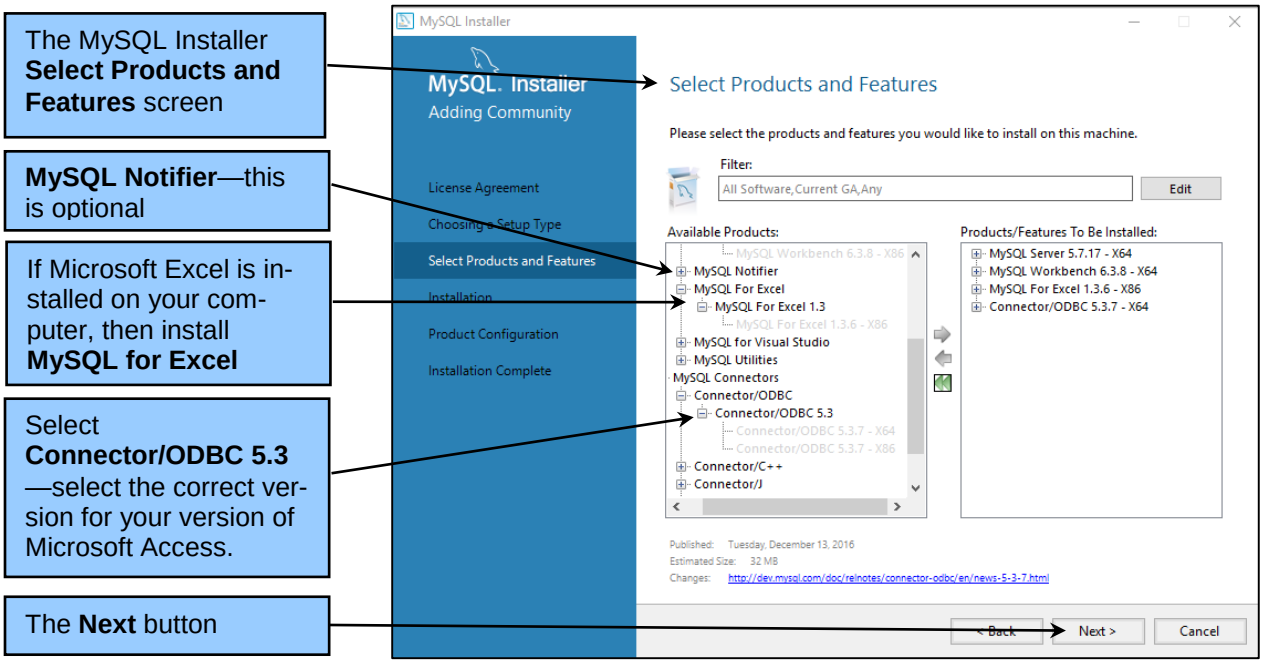


Figure C-1(e) — The MySQL Installer Select Products and Features Screen with Expanded Connectors

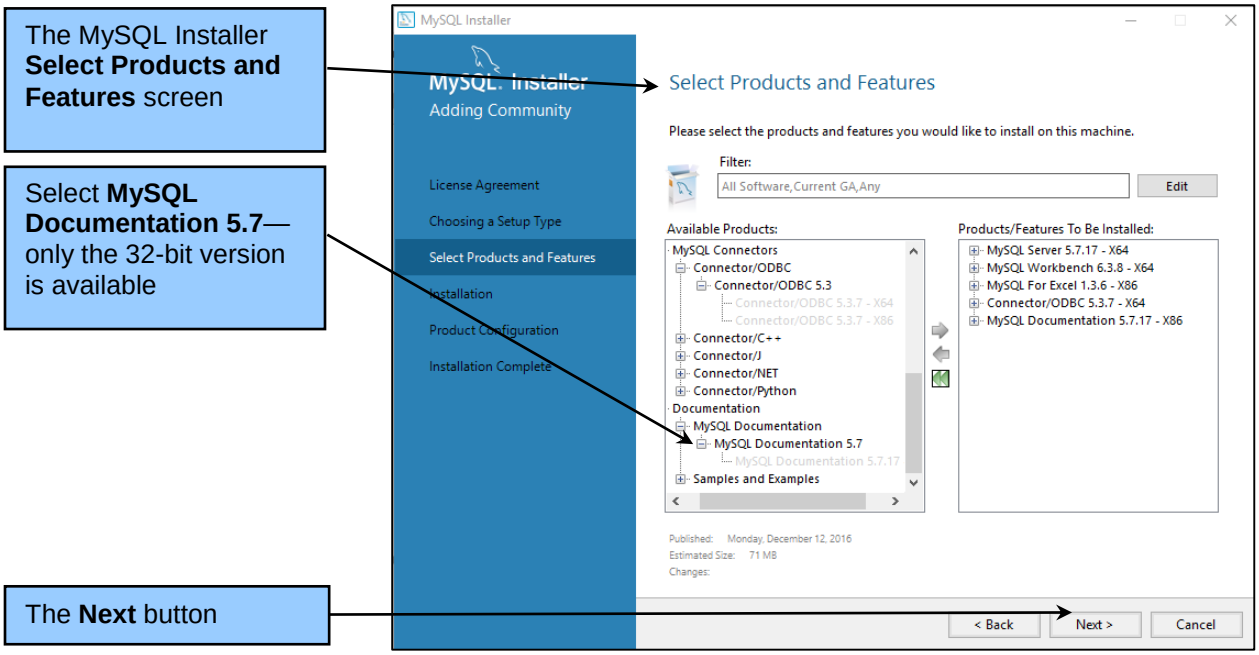


Figure C-1(f) — The MySQL Installer Select Products and Features Screen with Expanded Documentation

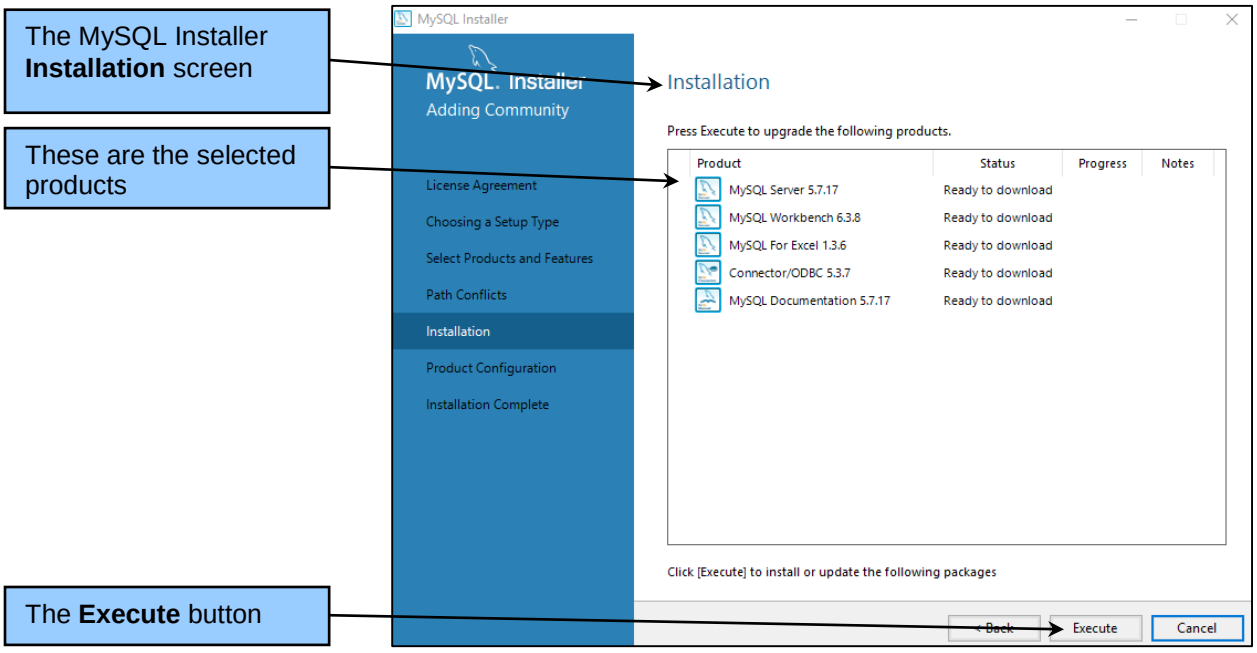


Figure C-1(g) — The MySQL Installer Installation Screen

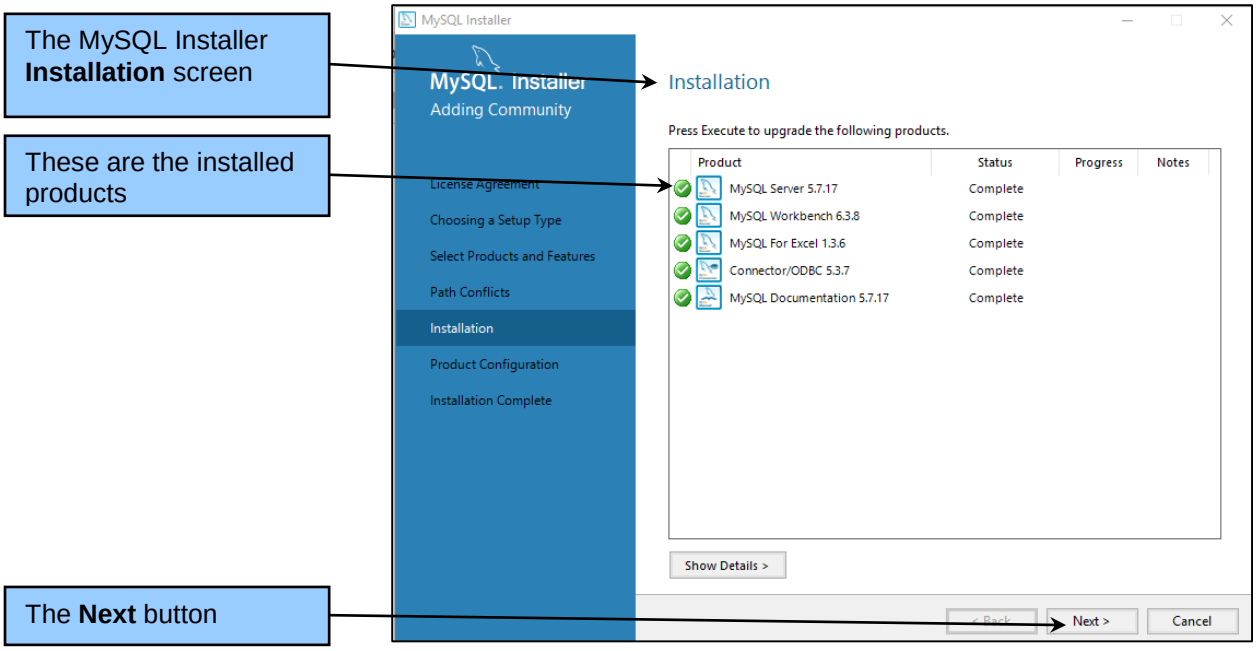


Figure C-1(h) — The MySQL Installer Installation Complete Screen

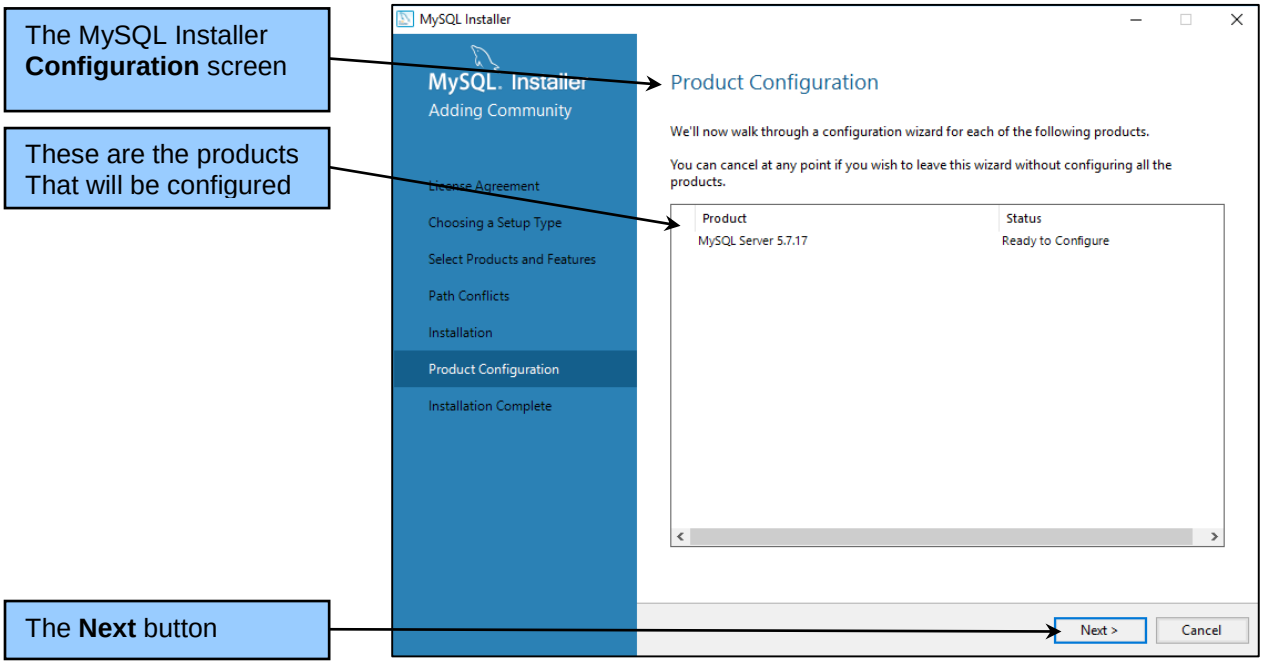


Figure C-1(i) — The MySQL Installer Product Configuration Screen

11. Click the **Next** button. The *Type and Networking* screen is displayed, as shown in Figure C-1(j). We will configure our installation of MySQL as the default development machine with the default TCP/IP options. We will not need to use the advanced configuration options.

12. Click the **Next** button. The *Accounts and Roles* screen is displayed, as shown in Figure C-1(k). We will need to enter a password for the *root user* account. The **root user** is the name of the administrator user in MySQL, and this user has complete administration privileges on the MySQL server. Choose a password, then enter it in both the **MySQL Root Password** and **Repeat Password** text boxes.

13. In order to add yourself as a MySQL server administrator, click the **Add User** button. The **MySQL User Detail** dialog box is displayed, as shown in Figure C-1(l). Enter your name as the Username, and enter a password. Leave the other settings as shown. When you are done, click the **OK** button to close the MySQL User Detail dialog box. We will need this user for the ODBC Connector section later and for Chapter 7.

14. Back in the *Accounts and Roles* screen, you will see yourself added as a user. Click the **Next** button to display the *Windows Service* screen. We will use the default settings on this screen, so click the **Next** button.

15. The *Plugins and Extensions* screen is displayed, which allows the option of adding in a new Document Database NoSQL extension using X Protocol. We are skipping it for now, so just click the **Next** button.

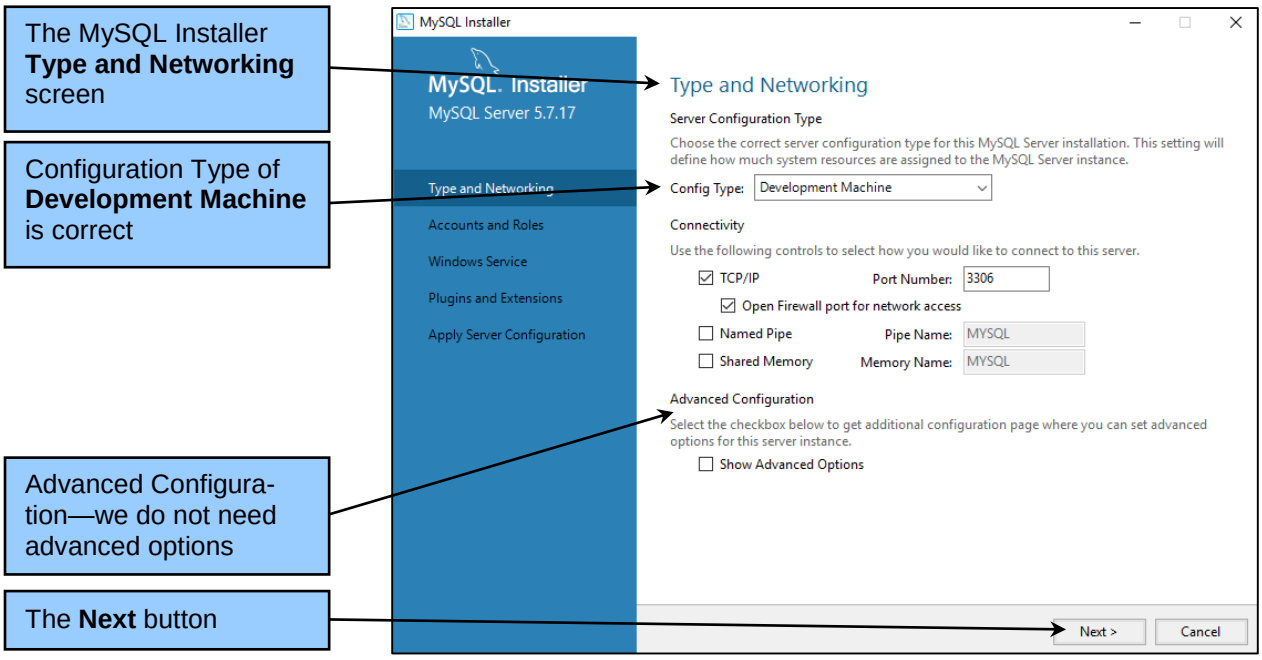


Figure C-1(j) — The MySQL Installer Type and Networking Screen

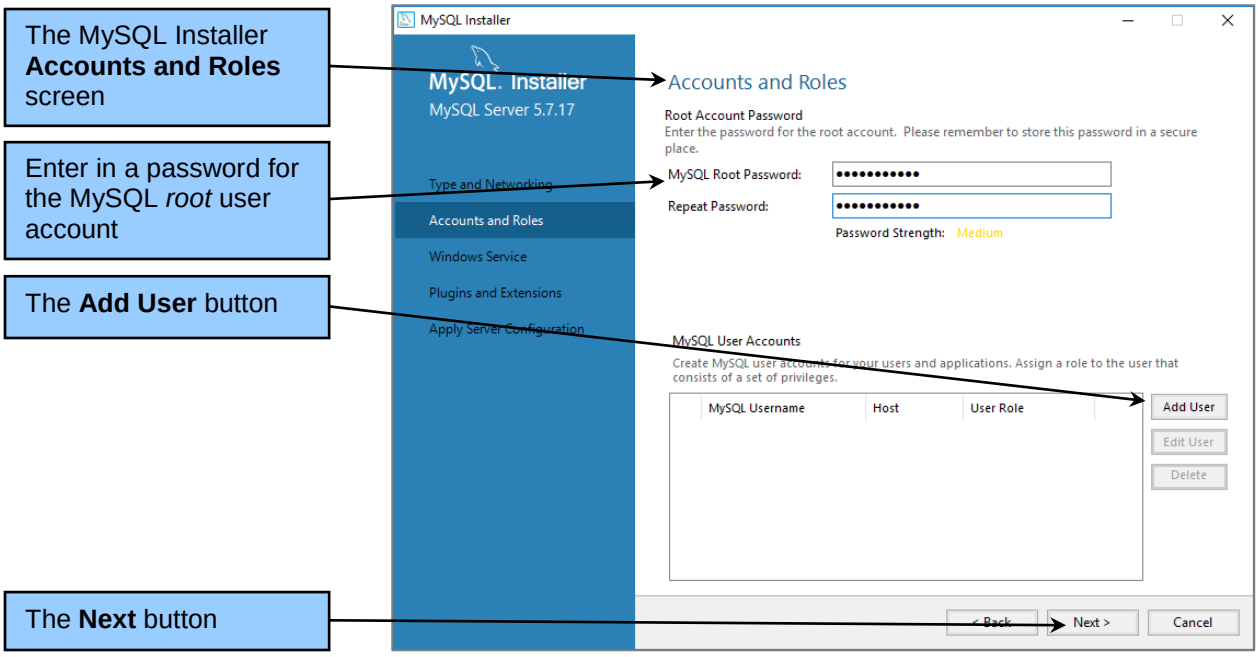


Figure C-1(k) — The MySQL Installer Accounts and Roles Screen

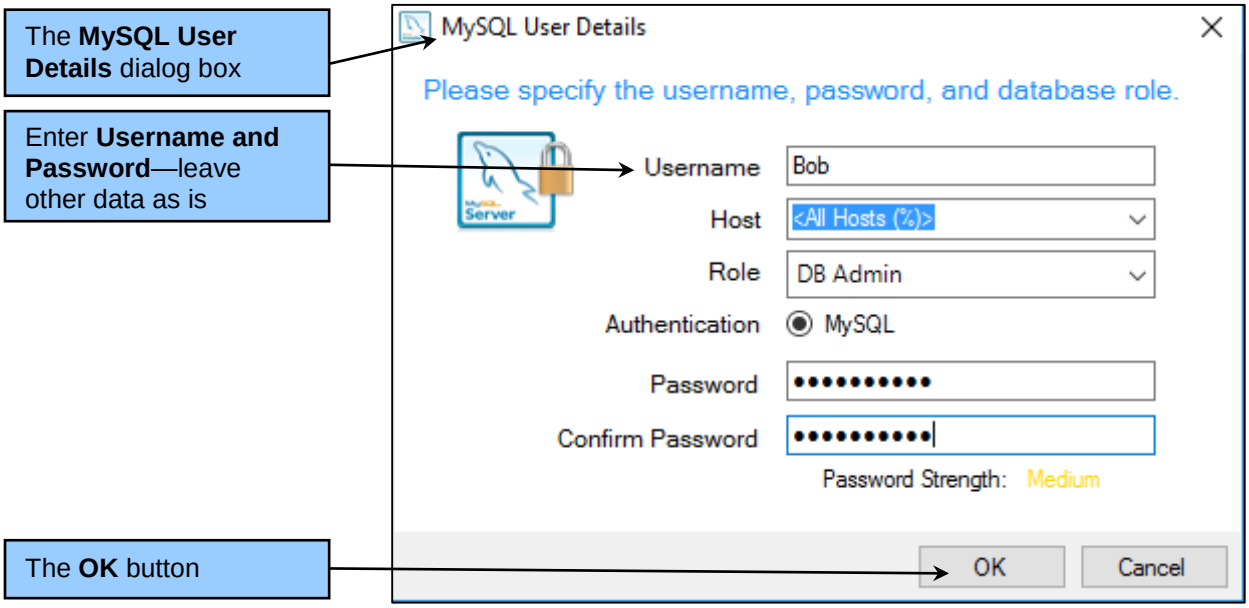


Figure C-1(l) — The MySQL Installer MySQL User Details Screen

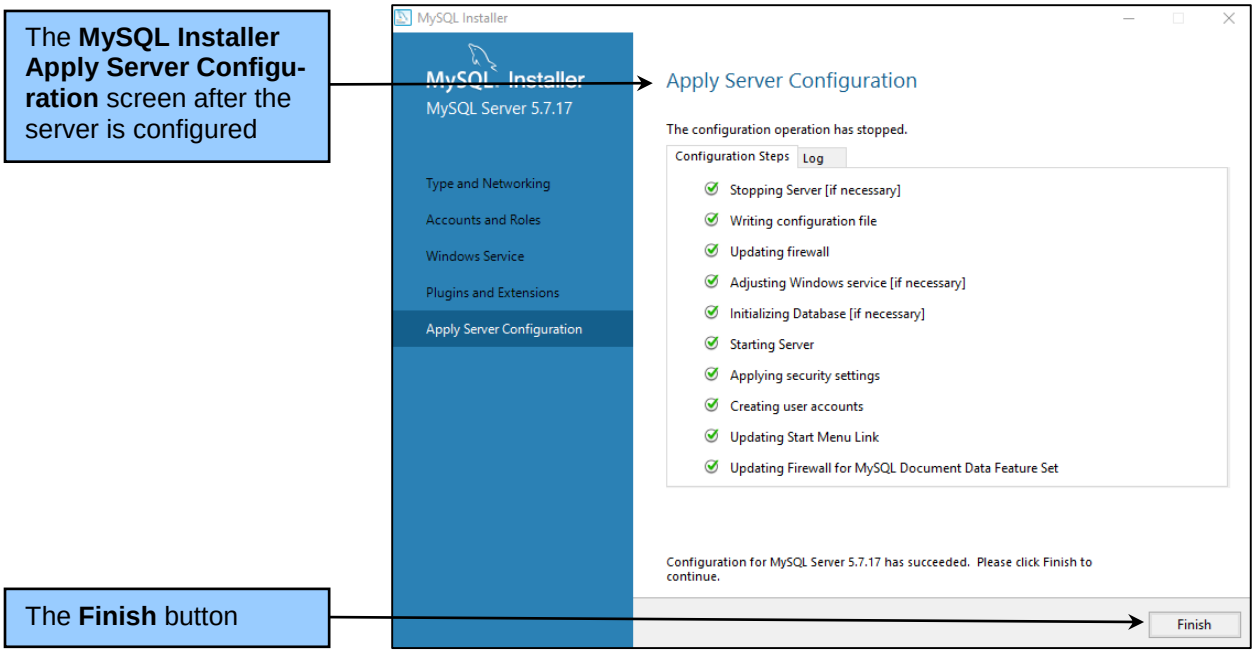


Figure C-1(m) — The MySQL Installer Apply Server Configuration Screen

16. The *Apply Server Configuration* screen is displayed, which summarizes the steps that will be run to configure the server. Click the **Execute** button to configure the MySQL server. When the configuration is complete, the *Apply Server Configuration* screen is updated and displayed as shown in Figure C-1(m).

17. Click the **Finish** button. The *Product Configuration* screen is displayed, showing that the configuration is complete in Figure C-1(n). Click the **Next** button.

18. The *Installation Complete* screen is displayed, with the *Start MySQL Workbench after Setup* checkbox checked, as shown in Figure C-1(o). Click the **Finish** button.

19. The MySQL Installer dialog box closes, and the MySQL Workbench is opened and displayed as shown in Figure C-2. The MySQL Workbench will be the main MySQL GUI utility we use for our work in MySQL 5.7.

20. Now is a good time to add the MySQL Workbench icon to the Windows Taskbar—this will make it much more convenient to open MySQL Workbench. Right-click the MySQL Workbench icon currently on the Taskbar, then click **Pin to taskbar**.

21. Click the MySQL Workbench **X [Close]** button to close MySQL Workbench. Now is a good time to add the MySQL Workbench icon to the Windows Task bar—this will make it much more convenient to open MySQL Workbench.

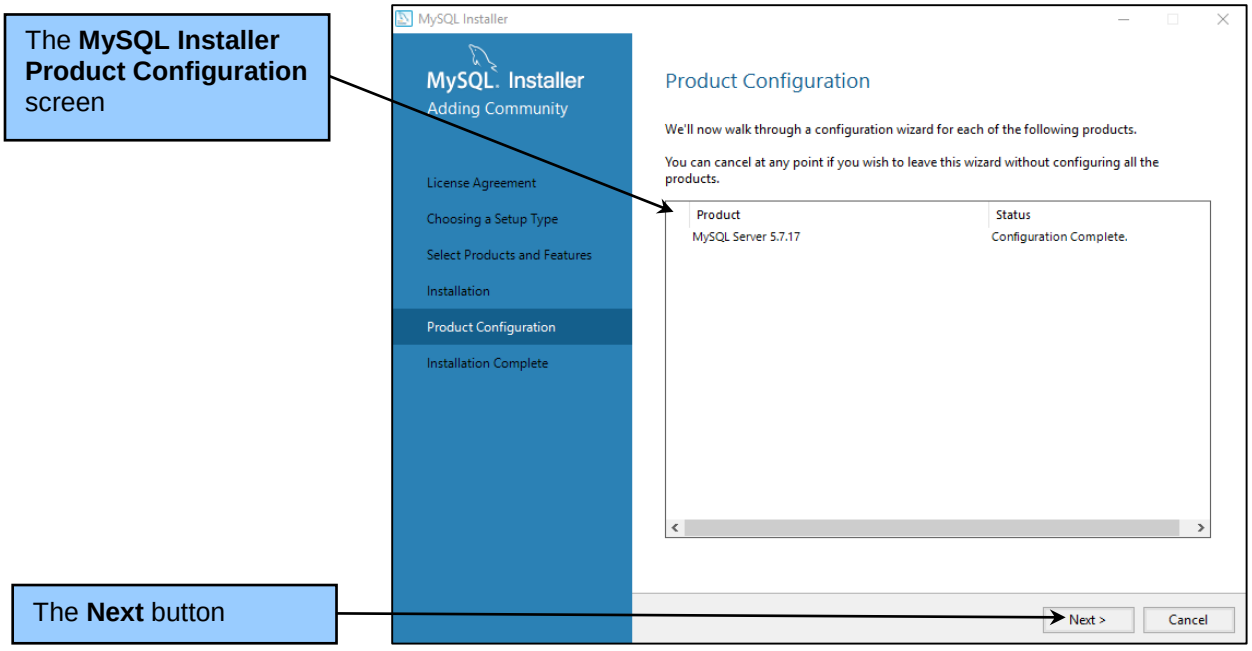


Figure C-1(n) — The MySQL Installer Product Configuration Screen After Configuration

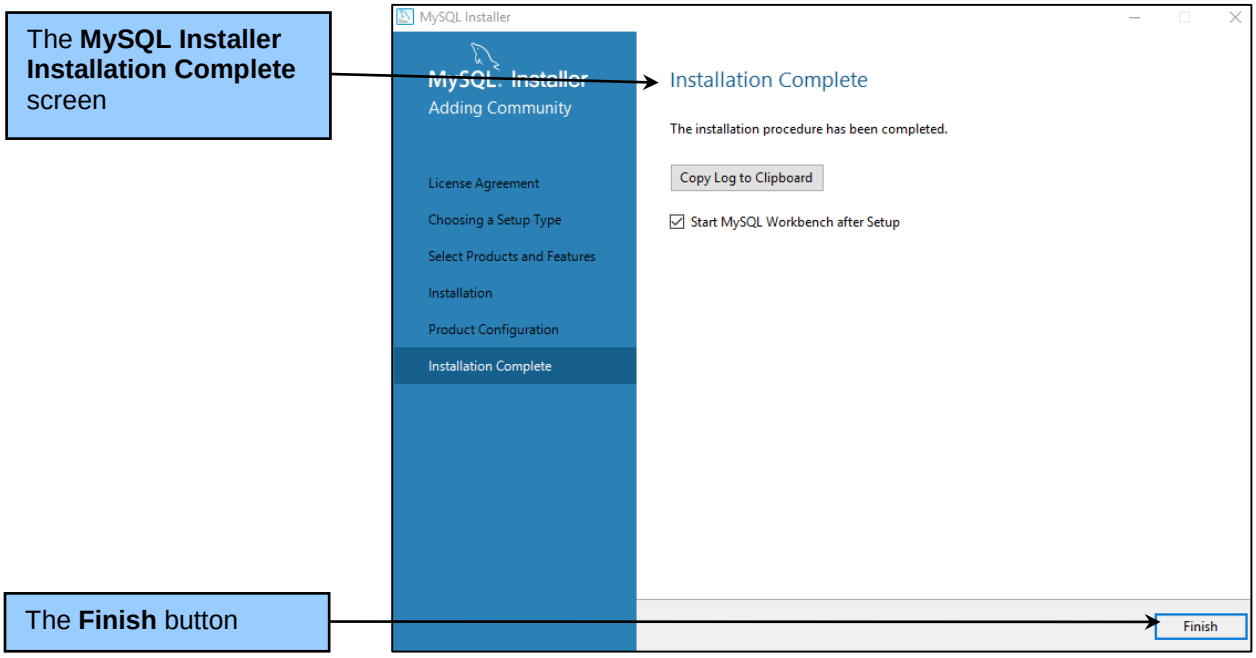


Figure C-1(o) — The MySQL Installer Installation Complete Screen

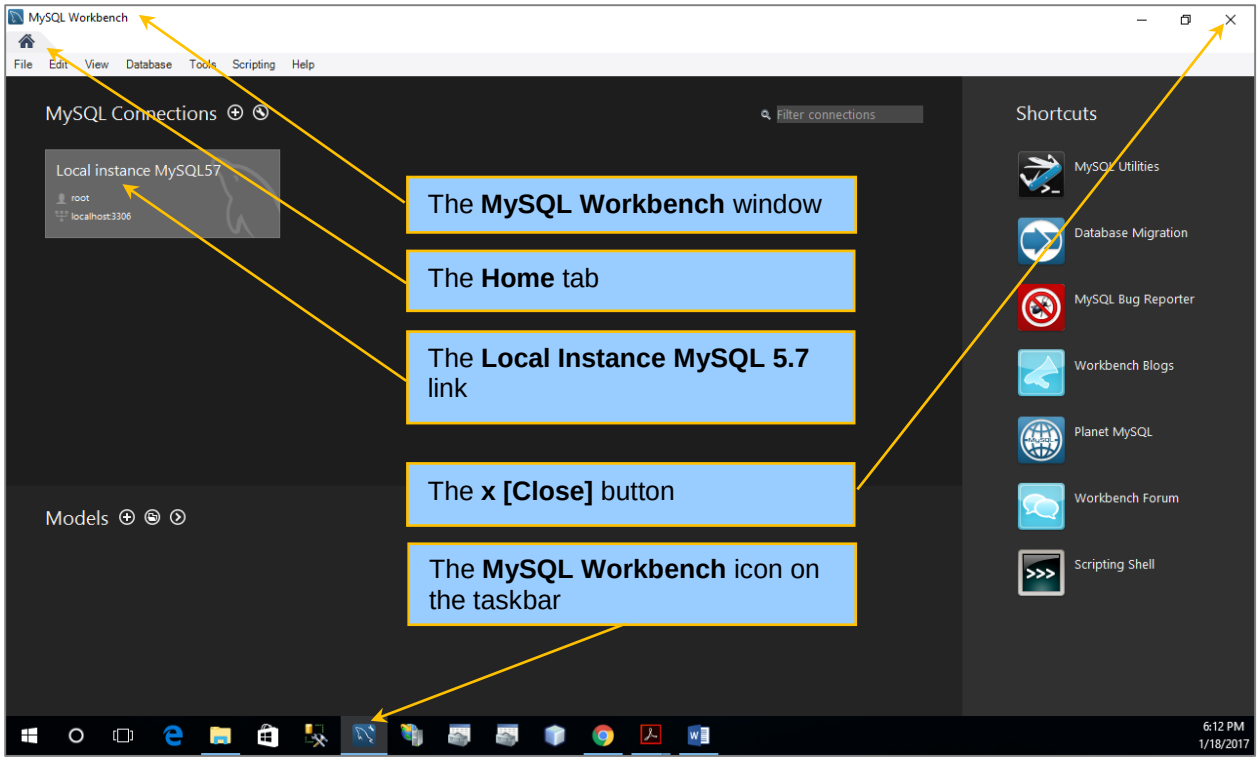


Figure C-2 — The MySQL Workbench

By The Way

Your installation of MySQL will install the current versions of these products, which may have different versions of the products shown here. That is OK, as you will want to have the most current versions of the software installed. The MySQL Windows Installer can be used to check for updates.

How Do I Create a Workspace for the MySQL Workbench Files?

Before using the MySQL Workbench we recommend creating a folder named **MySQL Workbench** under the **My Documents** folder (or whatever your main data storage area is named). In Windows, this can be done using Windows Explorer, as shown in Figure C-3. In this workspace, create a subfolder for each database project. In this workspace, create two folders, *EER Models* (for database designs) and *Schemas* (for database scripts). In the Schemas folder, create a subfolder for each database project each time you start a new database project.

How Do I Start the MySQL Workbench?

To start the MySQL Workbench running in Microsoft Windows 10, click the **MySQL Workbench 6.3 CE** icon.[5] The MySQL Workbench splash screen is displayed, followed by the MySQL Workbench window with the Home page displayed, as shown in Figure C-2.

The MySQL Workbench Home tab is a dashboard allowing us access to MySQL database design (mislabeled as "Models"), SQL database development and MySQL DBMS administration. There are several means of accessing the features provided, and we will only cover some basic ones here.[6]

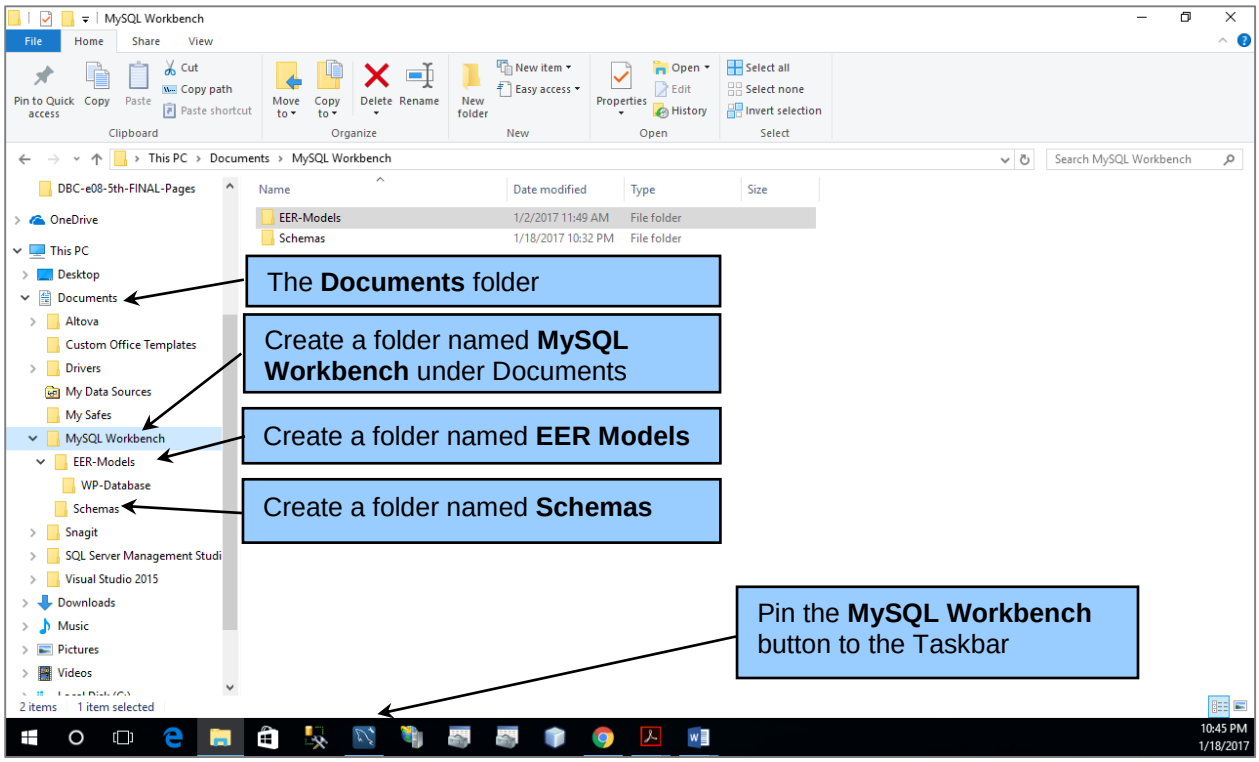


Figure C-3 — The MySQL Workbench Folder in Windows Explorer

[5] Alternatively, we recommend that you pin MySQL Workbench 6.3 CE to the Taskbar, as shown in Figure C-3, and start the MySQL Workbench from this icon. The *CE* designation shows that this is the open source, freely downloadable community edition—there is also a commercial version available that includes additional functionality.

[6] If any models are listed in the Models section, right-click each model name and click Remove Model File from List.

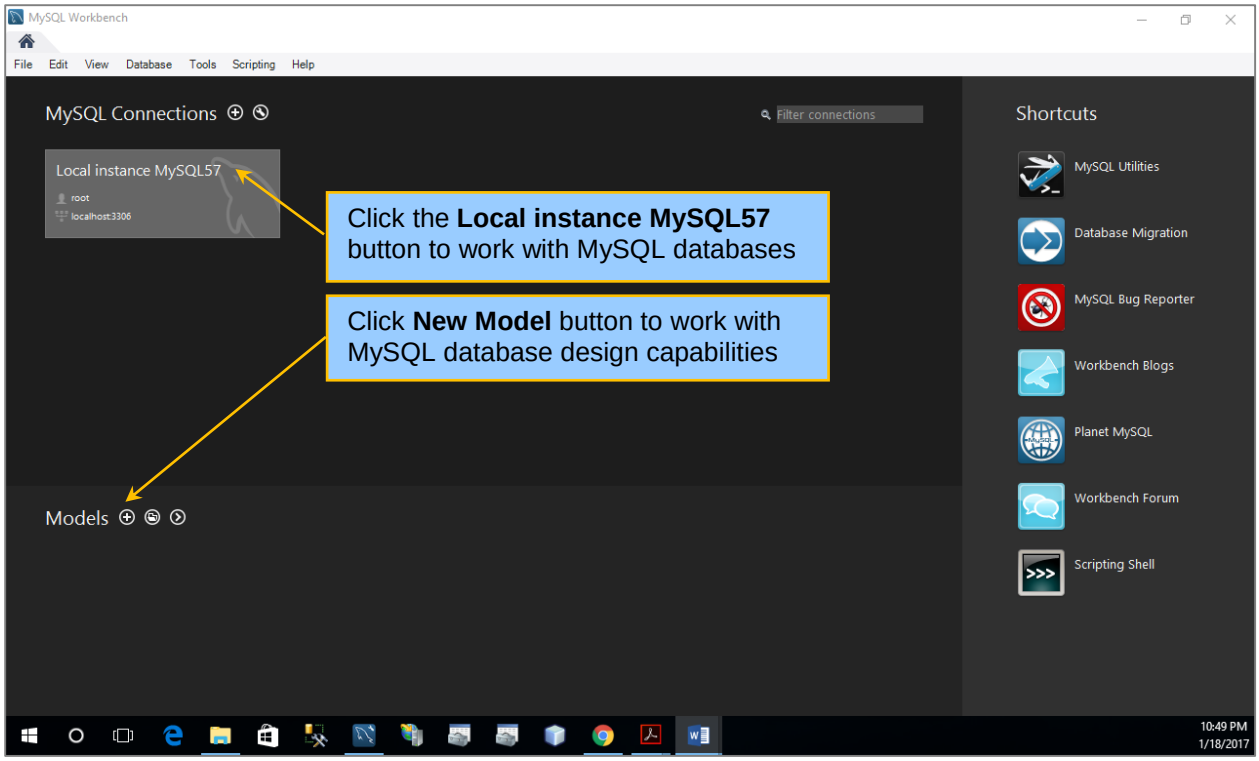


Figure C-4 — The MySQL Workbench Home Tab **with the Local instance MySQL57 button**

How Do I Create a Database in MySQL?

To create a database in MySQL Workbench, start by clicking the **Local instance MySQL57** button in the **SQL Connections** section of the MySQL Workbench Home page, as shown in Figure C-4. The **Connect to MySQL Server** dialog box is displayed, as shown in Figure C-5. Enter the password for the Root user, and then click the **OK** button. The MySQL Workbench reappears, but now with an open tabbed connection window named Local Instance MySQL57, as shown in Figure C-6.

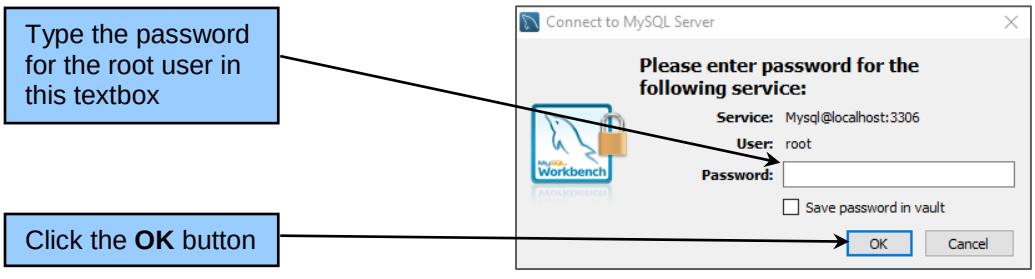


Figure C-5 — The Connect to MySQL Server Dialog Box

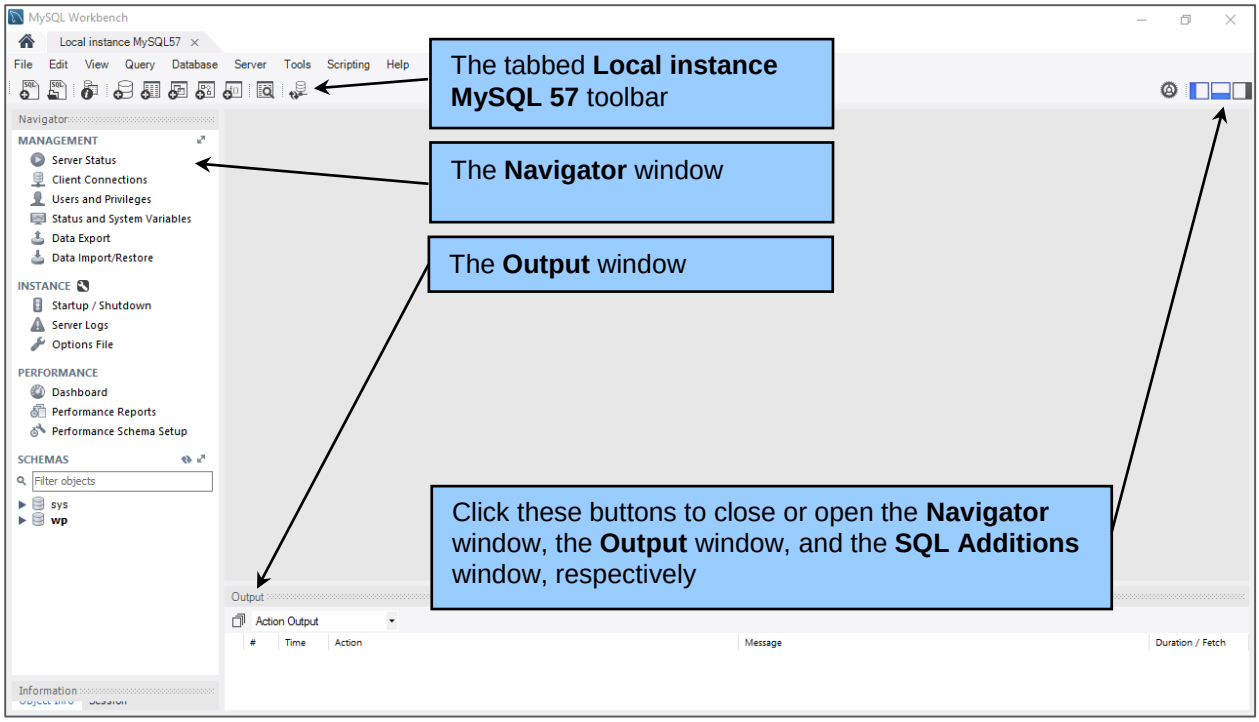


Figure C-6 — The Navigator and the Other Windows

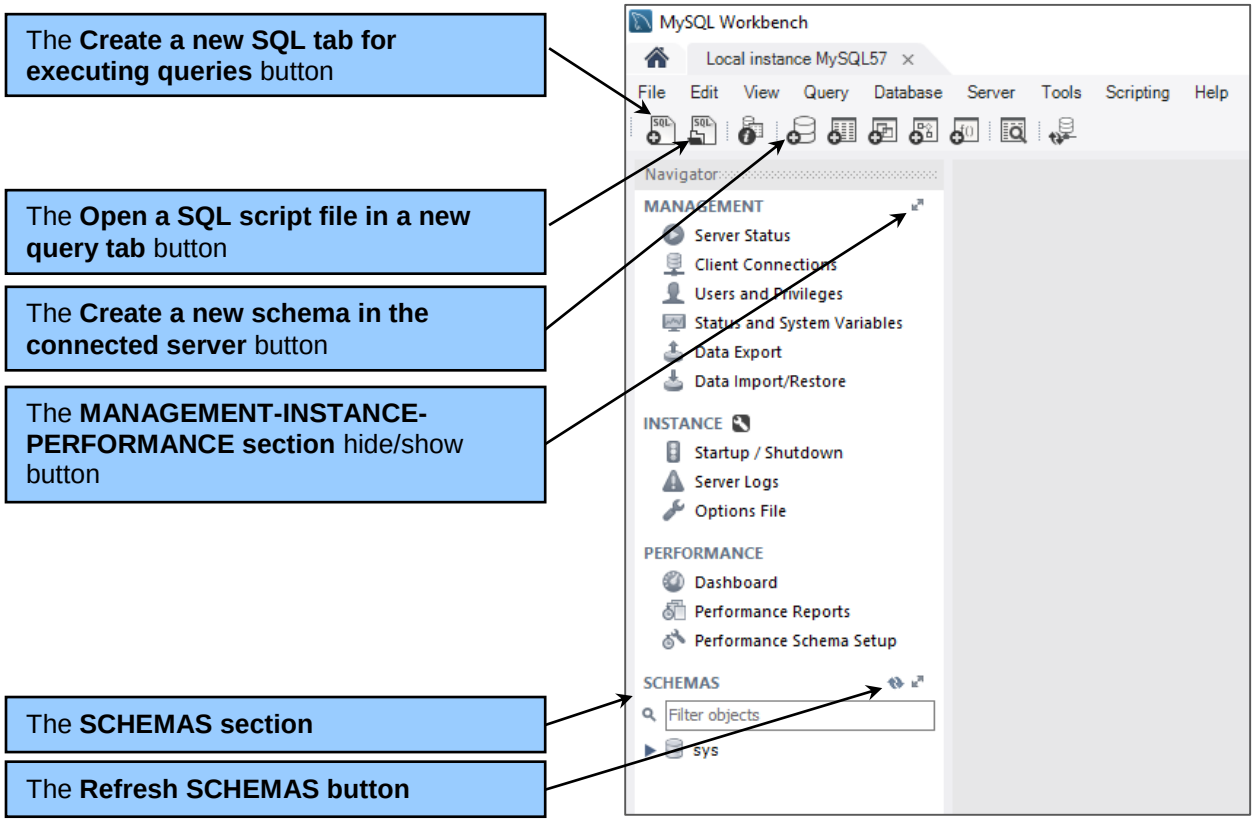


Figure C-7 — The Navigator Window and Toolbars

In Figure C-6, we can see some of the features of the tabbed connection window:

1. The **Navigator**, which displays management functions and existing schemas—**Schema** is MySQL's term for a database. Note only the *sys* schema (automatically created by MySQL) exists.
2. The **Output** window—Test messages will be displayed here as we work on databases.
3. The **SQL Additions** window—We can store SQL code snippets (short bits of SQL code) here for reuse.
4. The three buttons that control the display of the Object Browser, the Output window, and the SQL Additions window.

In Figure C-7, we can see some features of the Navigator window:

1. The **MANAGEMENT-INSTANCE-PERFORMANCE section**, which provide the management functions for the schemas.
2. The **MANAGEMENT-INSTANCE-PERFORMANCE section** hide/show button will hide the management functions, and the button in the SCHEMAS section is used to show the functions if they are hidden.
3. The **Create a new SQL tab for executing queries** button—we will use this to open a new SQL editor tab.
4. The **Open a SQL script file in a new query tab** button—we will use this to open an existing SQL script file.
5. The **Create a new schema in the connected server** button—it is used to create a new database (schema).
6. The **SCHEMAS section**—This section is where we have control of the MySQL schemas.

We will not use the MANAGEMENT-INSTANCE-PERFORMANCE section functionality nor the SQL Additions window shown in Figure C-6, so we will remove them from the display.

Controlling the MySQL Workbench SQL Editor Display:

1. Click the MANAGEMENT-INSTANCE_PERFORMANCE hide/show button (see the callout in Figure C-7). The MANAGEMENT-INSTANCE_PERFORMANCE section is no longer displayed, and the SCHEMA section expands into this display area.
2. If necessary, click the button that controls the SQL Additions window (see the last callout in Figure C-6). The SQL Additions window is no longer displayed, and the SQL File window expands into the display area previously used by the SQL Additions window.

Now we will create the Wedgewood Pacific (WP) database that we used as our example database in Chapter 3. Appendix E also uses the WP database.

Creating a MySQL Database:

1. Click the **Create a new schema in the connected server** button shown in Figure C-7.
2. The new_schema tabbed window is displayed, as shown in Figure C-8.
3. Type the new schema (database) name WP in the **Name** textbox, as shown in Figure C-9, and then click the **Apply** button. A warning dialog box will declare that the schema name will be in lower case letters as "wp." Click OK.
4. As shown in Figure C-10, an Apply SQL Script to Database dialog box is displayed so that the user can review the SQL command before it is executed. Click the **Apply** button.
5. As shown in Figure C-11, the Apply SQL Script to Database dialog box is displayed with the results of executing the SQL command. Click the **Finish** button.
6. The WP (the renamed *new_schema*) dialog box is displayed again in Figure C-12. Click the **X (Close)** button on the wp Schema tab.
7. The WP schema object now appears in the Object Browser. Note that although we entered the schema name as *WP* (all upper case), MySQL insists on displaying it as *wp* (all lower case). If you want to make sure the Object Browser objects are displayed correctly, then click **Refresh SCHEMAS** button (as shown in Figure C-7) which will refresh the display in the Object Browser.

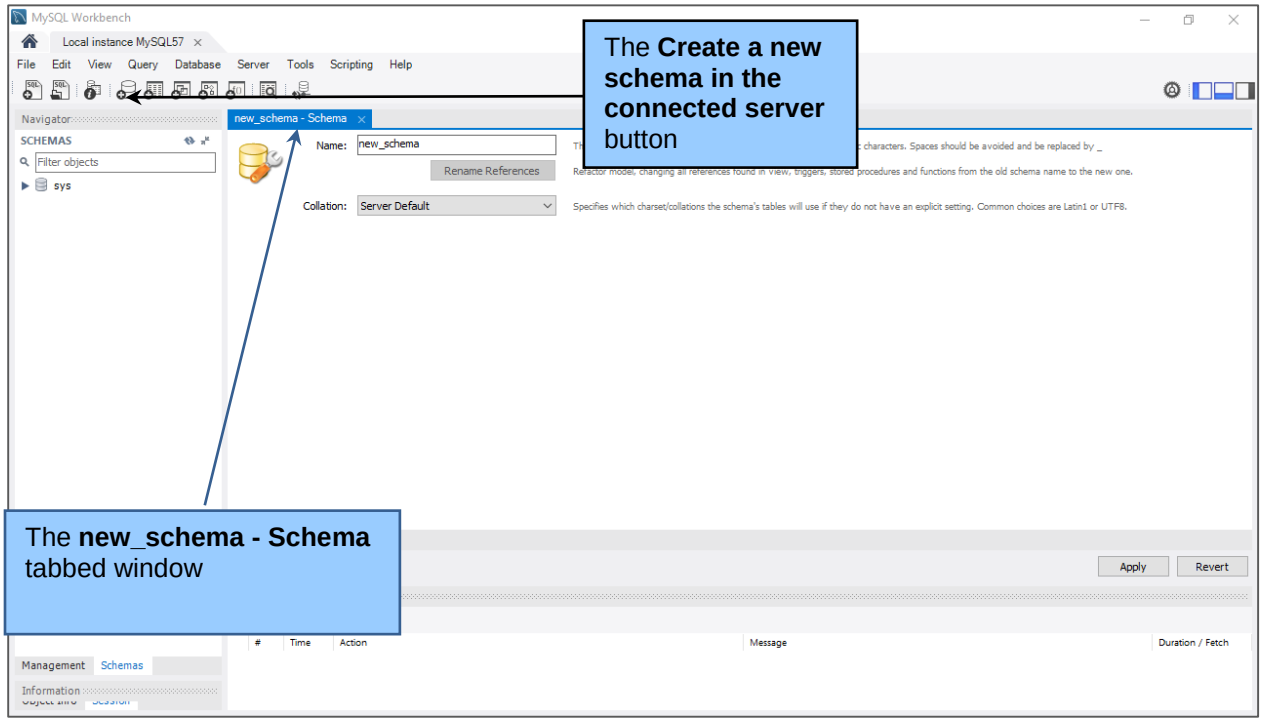


Figure C-8 — The new_schema Tabbed Window

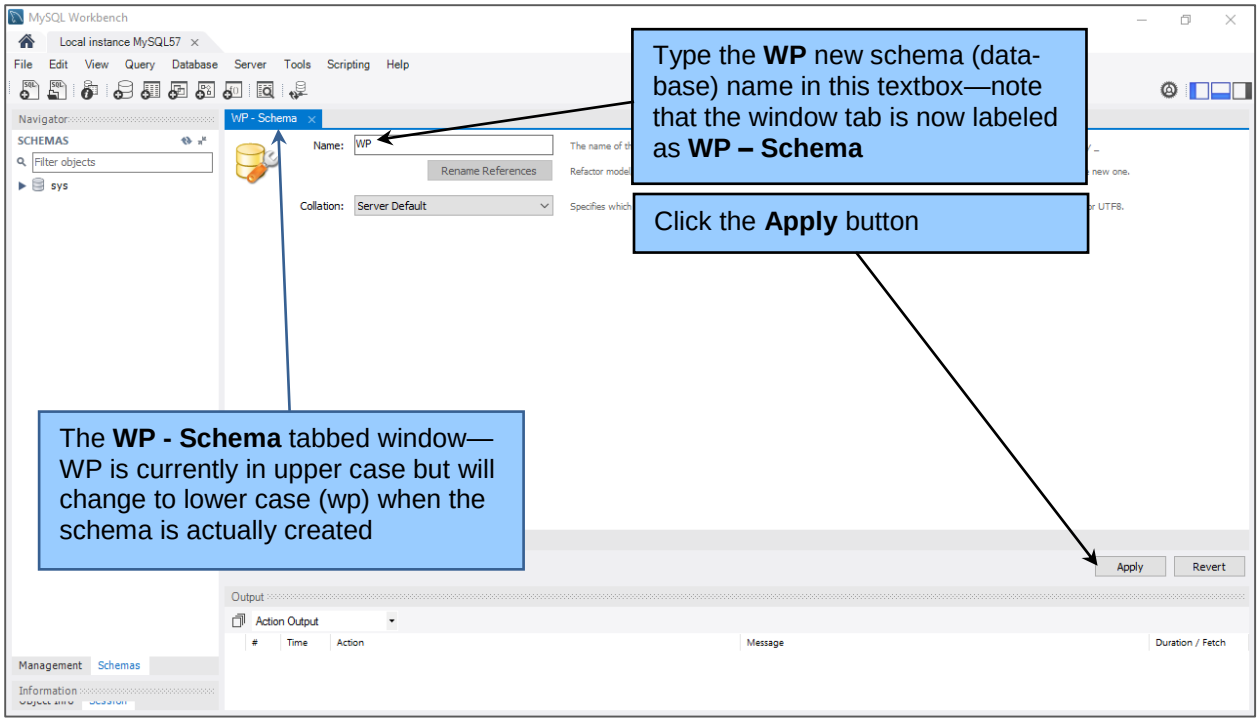


Figure C-9 — The WP Schema Name in the new_schema Dialog Box

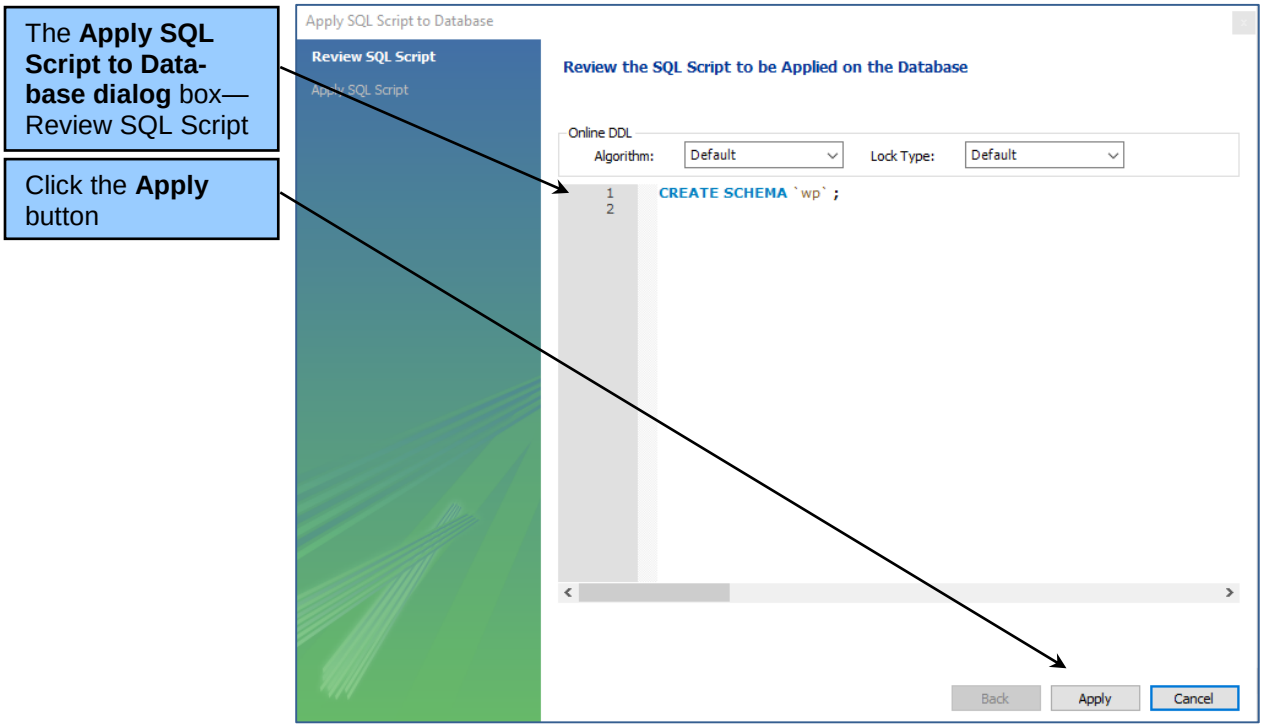


Figure C-10 — The Apply SQL Script to Database – Review the SQL Script to be Applied on the Database Dialog Box

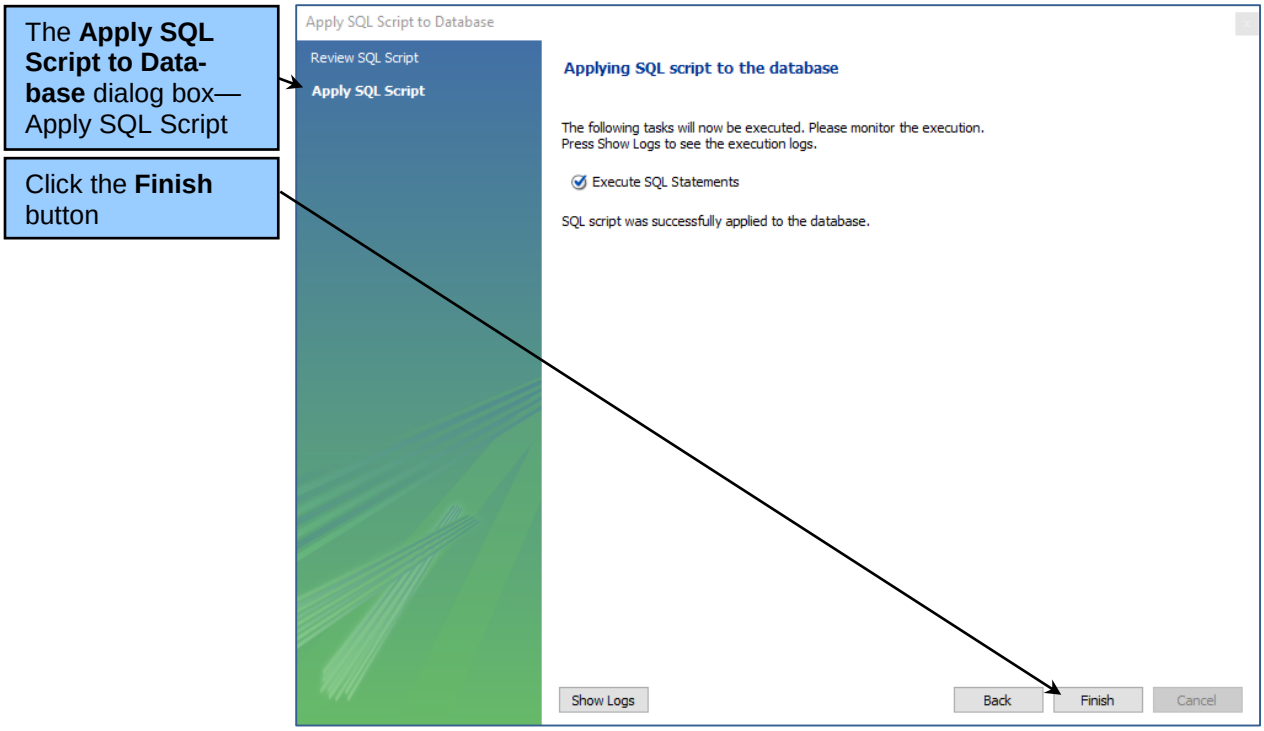


Figure C-11 — The Apply SQL Script to Database – Applying SQL Script Dialog Box

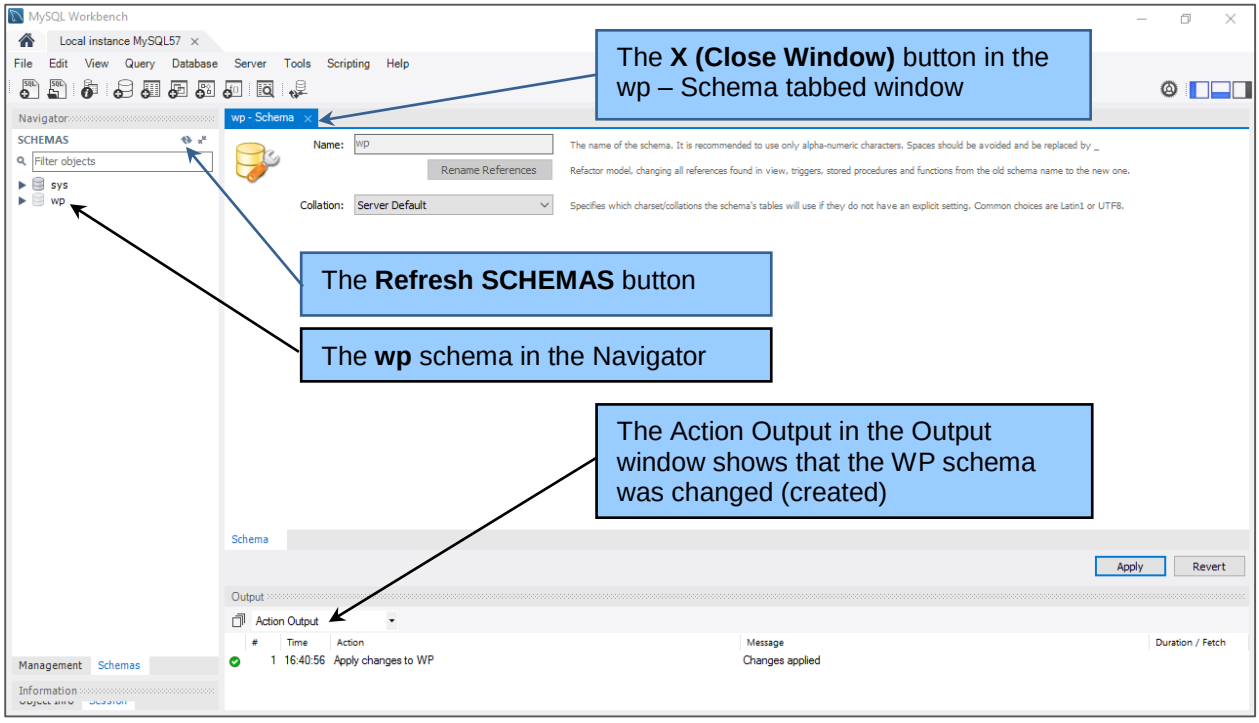


Figure C-12 — The wp Schema (Database) in the Object Browser

How Do I Set the Active Database in MySQL?

To work with a MySQL database, you must select it as the active database. In MySQL terms, this is called the **default schema**. To set the default schema, select the name of the database that you want to work with in the Default Schema drop-down list.

Setting the Default Schema:

1. Click the **wp** schema object in the object browser to select it.
2. Right-click the **wp** schema object to display the shortcut menu, as shown in Figure C-13.
3. In the shortcut menu, click the **Set as Default Schema** command.
4. The **wp** schema is set as the active schema (default schema), and the **wp** schema object is displayed in bold text in the Object Browser to indicate that it is the active schema, as shown in Figure C-14.

How Do I Work with SQL Statements in MySQL?

In MySQL, SQL statements can be run individually or as part of a related group of SQL statements known as an **SQL script**. Scripts, which are named with the ***.sql** extension, are efficient for processing groups of SQL statements. For example, you could create a set of CREATE TABLE commands to build a new database structure as a script, or you could create a set of INSERT commands to use when data need to be added to a table as a script.

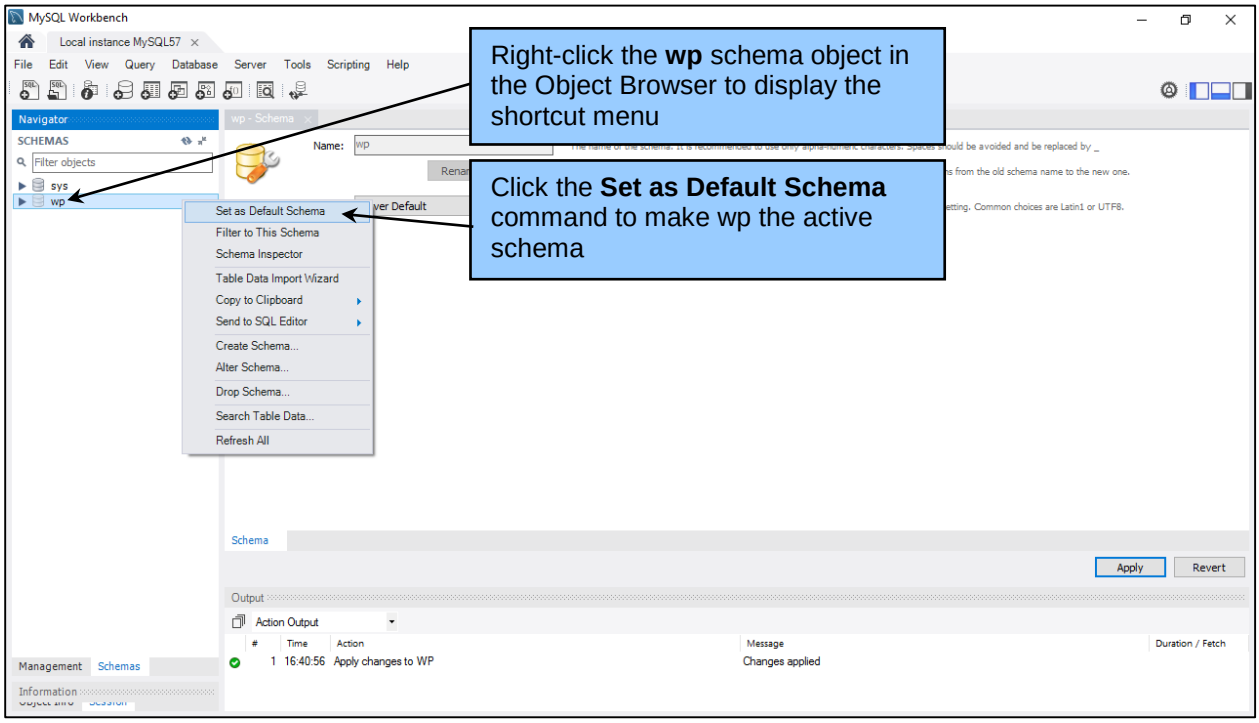


Figure C-13 — The Set as Default Schema Command

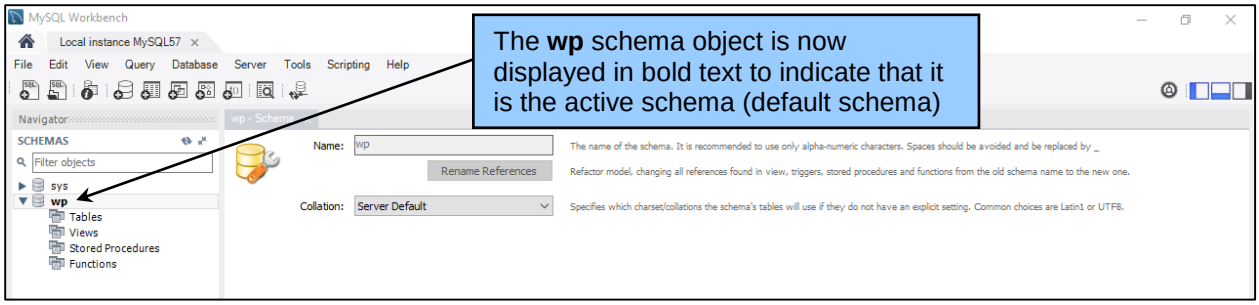


Figure C-14 — The WP Schema as the Active Schema (Default Schema)

SQL Commands to Create Table Structures

To open the SQL File tabbed window, click on the **Create a new SQL tab for executing queries** button, as shown in Figure C-15. This opens the script editor and, in fact, is the preferred script editor when we are working with MySQL databases. New buttons to execute and save scripts are now shown. Figure C-16 shows the MySQL version of the SQL statements to create the WP database shown in Figure 3-7. The major MySQL data types are shown in Chapter 3 in Figure 3-5(c). Comparing the two versions of these SQL statements will help you understand any differences in how SQL is used in MySQL. Enter (or copy and paste) the statement in C-16 and C-16 (continued) below in the query window, then save them as *DBC-e08-MySQL-WP-Create-Tables.sql*. The same statements are shown in MySQL Workbench in Figure C-17, together with some added header comments to document the script.

By The Way

MySQL uses the **AUTO_INCREMENT** keyword to set surrogate key values. The default values for AUTO_INCREMENT are an initial value of 1 and an increment of 1. After the table is created, you can modify the initial value by using the ALTER TABLE statement. For example, in the WP database, the PROJECT table has a surrogate key that starts at 1000 and increments by 100. You could set the correct initial value by using the SQL statement:

```
ALTER TABLE PROJECT AUTO_INCREMENT=1000;
```

Unfortunately, you cannot change the MySQL increment of 1. So for cases such as the ProjectID values in PROJECT, we will manually insert the correct values in MySQL.

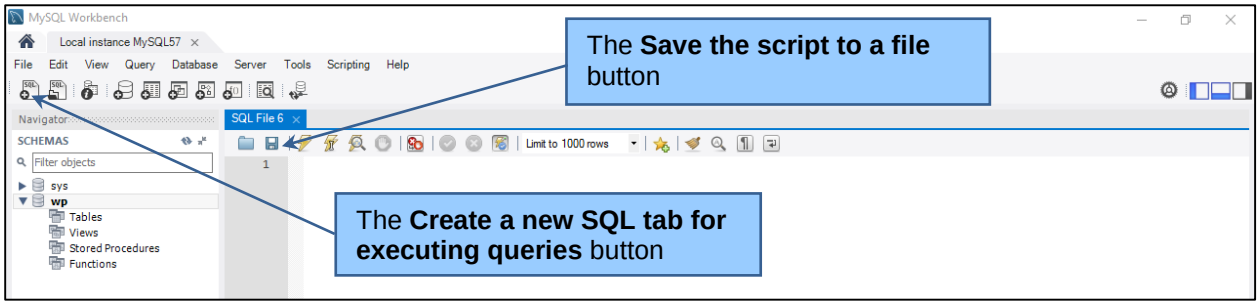


Figure C-15 — Create a New SQL Tab for Executing Queries

```
/*************************************************************************************/
/*                                                                                   */
/*     Kroenke and Auer - Database Concepts (8th Edition) Chapter 03                 */
/*                                                                                   */
/*     Wedgewood Pacific [WP] Create Tables                                          */
/*                                                                                   */
/*     These are the MySQL 5.7 SQL code solutions                                    */
/*                                                                                   */
/*     NOTE: ALTER TABLE statements can be used to set the                           */
/*     MySQL AUTO_INCREMENT to a starting value other than 1. For example:           */
/*           ALTER TABLE PROJECT AUTO_INCREMENT=1000;                                */
/*     However, the increment is always one (1), so PROJECT ProjectID                */
/*     values must be manually inserted                                              */
/*                                                                                   */
/*************************************************************************************/

CREATE  TABLE DEPARTMENT(
       DepartmentName      Char(35)            NOT NULL,
       BudgetCode          Char(30)            NOT NULL,
       OfficeNumber        Char(15)            NOT NULL,
       DepartmentPhone     Char(12)            NOT NULL,
       CONSTRAINT          DEPARTMENT_PK       PRIMARY KEY(DepartmentName)
       );

CREATE  TABLE EMPLOYEE(
       EmployeeNumber      Int                 NOT NULL AUTO_INCREMENT,
       FirstName           Char(25)            NOT NULL,
       LastName            Char(25)            NOT NULL,
       Department          Char(35)            NOT NULL DEFAULT 'Human Resources',
       Position            Char(35)            NULL,
       Supervisor          Int                 NULL,
       OfficePhone         Char(12)            NULL,
       EmailAddress        VarChar(100)        NOT NULL UNIQUE,
       CONSTRAINT          EMPLOYEE_PK         PRIMARY KEY(EmployeeNumber),
       CONSTRAINT          EMP_DEPART_FK       FOREIGN KEY(Department)
                                 REFERENCES DEPARTMENT(DepartmentName)
                                       ON UPDATE CASCADE,
       CONSTRAINT          EMP_SUPER_FK        FOREIGN KEY(Supervisor)
                                 REFERENCES EMPLOYEE(EmployeeNumber)
       );

CREATE  TABLE PROJECT(
       ProjectID           Int                 NOT NULL,
       ProjectName         Char(50)            NOT NULL,
       Department          Char(35)            NOT NULL,
       MaxHours            Decimal(8,2)        NOT NULL DEFAULT 100,
       StartDate           Date                NULL,
       EndDate             Date                NULL,
       CONSTRAINT          PROJECT_PK          PRIMARY KEY (ProjectID),
       CONSTRAINT          PROJ_DEPART_FK      FOREIGN KEY (Department)
                                 REFERENCES DEPARTMENT(DepartmentName)
                                       ON UPDATE CASCADE
       );
```

Figure C-16 — The WP Database Create Table SQL Statements for MySQL

```
CREATE  TABLE ASSIGNMENT (
       ProjectID            Int                  NOT NULL,
       EmployeeNumber       Int                  NOT NULL,
       HoursWorked          Numeric(6,2)         NULL,
       CONSTRAINT           ASSIGNMENT_PK        PRIMARY KEY(ProjectID, EmployeeNumber),
       CONSTRAINT           ASSIGN_PROJ_FK       FOREIGN KEY (ProjectID)
                                  REFERENCES PROJECT (ProjectID)
                                         ON UPDATE NO ACTION
                                         ON DELETE CASCADE,
       CONSTRAINT           ASSIGN_EMP_FK        FOREIGN KEY (EmployeeNumber)
                                  REFERENCES EMPLOYEE(EmployeeNumber)
                                         ON UPDATE NO ACTION
                                         ON DELETE NO ACTION
       );

/*****************************************************************************/
```

Figure C-16 — The WP Database Create Table SQL Statements for MySQL (continued)

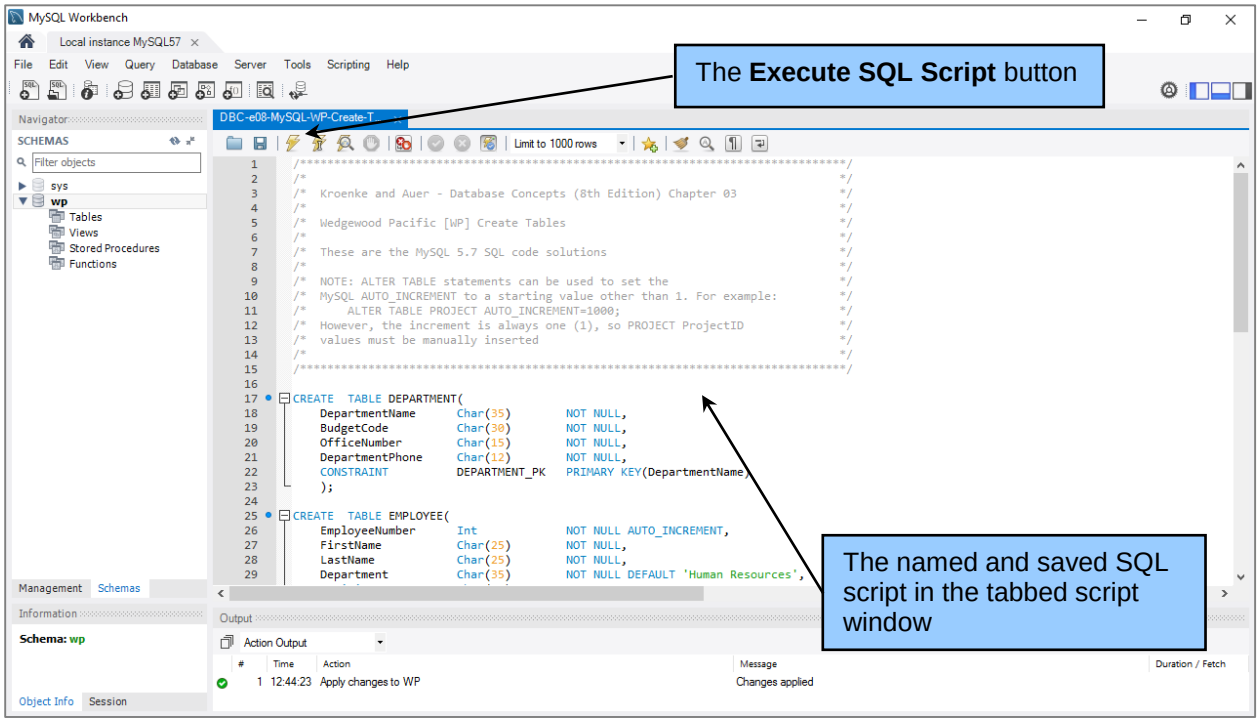


Figure C-17 — The WP Script in the MySQL Workbench

How Do I Execute an SQL Script in MySQL Workbench?

After the script has been named and saved (it is named during the save), make sure that WP is the active database, and then click the **Execute SQL Script** button (the *thunderbolt* icon) to run the entire script. (Note: The icon next to it will only execute the SQL statement under the cursor.) After the script executes, you can expand the WP schema object to display the tables created by the script. To close the Script window, click the **X** symbol on the Script window tab. The results are shown in Figure C-18.

By The Way

Believe it or not, the full name of the button that we have labeled as the *Execute SQL Script* button is the **Execute the selected portion of the script or everything, if there is no selection** button, according to the mouse roll-over tool tip. To see this for yourself, move the mouse over the button and leave it there for a few seconds. A tool tip window will open with the full name of the button. The full name is very descriptive and accurate but a bit long to be used in our instructions!

SQL Commands to Insert Data

Figure C-19 (and its continuations) show the MySQL version of the SQL statements to populate the WP database with data shown in Figure 3-12 (but we added the ProjectID values in the data, per our discussion of AUTO_INCREMENT). Comparing the two versions of these SQL statements will help you understand any differences in how SQL is used in MySQL. Enter (or cut and paste) the SQL statements to create and save an SQL script named *DBC-e08-MySQL-WP-Insert-Data.sql* based on Figure C-19, and then execute the script to populate the WP database.

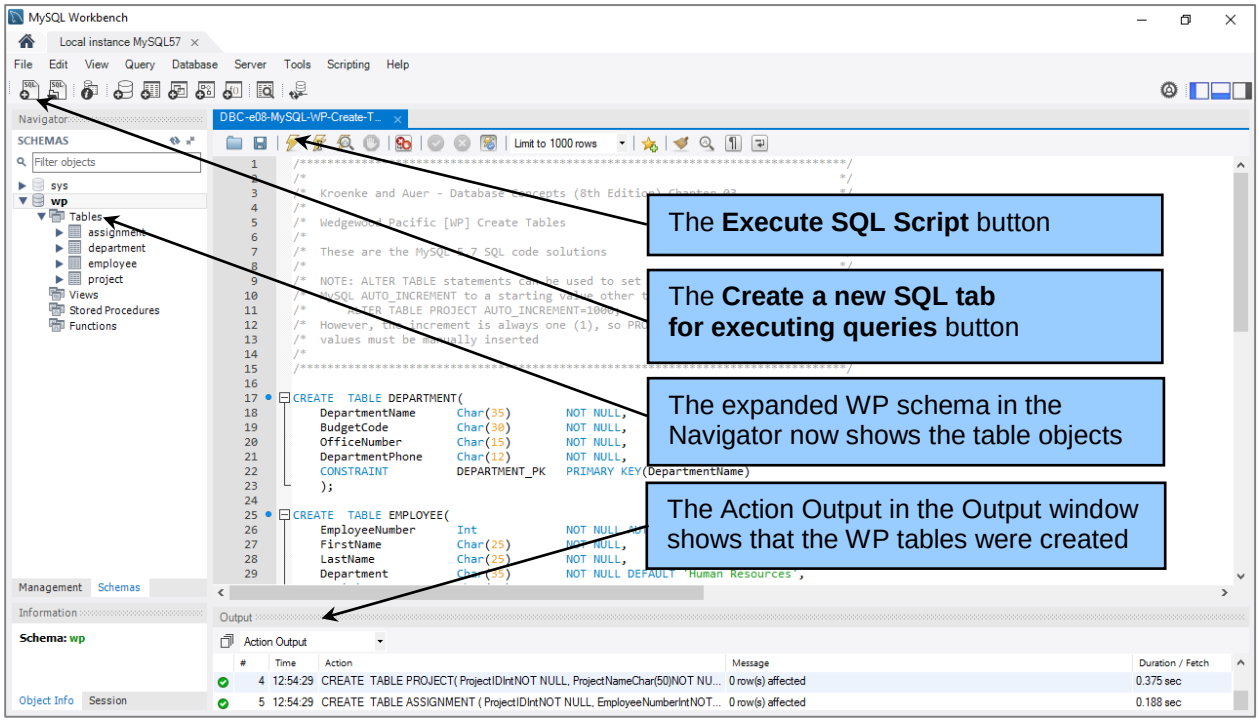


Figure C-18 — The SQL Script Results

```
/*********************************************************************************/
/*                                                                               */
/*     Kroenke and Auer - Database Concepts (8th Edition) Chapter 03             */
/*                                                                               */
/*     Wedgewood Pacific [WP] Database Data from Figure 3-12                     */
/*                                                                               */
/*     These are the MySQL 5.7 SQL code solutions                                */
/*                                                                               */
/*********************************************************************************/
/***** DEPARTMENT DATA ***************************************************/

INSERT INTO DEPARTMENT VALUES(
       'Administration', 'BC-100-10', 'BLDG01-210', '360-285-8100');
INSERT INTO DEPARTMENT VALUES(
       'Legal', 'BC-200-10', 'BLDG01-220', '360-285-8200');
INSERT INTO DEPARTMENT VALUES(
       'Human Resources', 'BC-300-10', 'BLDG01-230', '360-285-8300');
INSERT INTO DEPARTMENT VALUES(
       'Finance', 'BC-400-10', 'BLDG01-110', '360-285-8400');
INSERT INTO DEPARTMENT VALUES(
       'Accounting', 'BC-500-10', 'BLDG01-120', '360-285-8405');
INSERT INTO DEPARTMENT VALUES(
       'Sales and Marketing', 'BC-600-10', 'BLDG01-250', '360-287-8500');
INSERT INTO DEPARTMENT VALUES(
       'InfoSystems', 'BC-700-10', 'BLDG02-210', '360-287-8600');
INSERT INTO DEPARTMENT VALUES(
       'Research and Development', 'BC-800-10', 'BLDG02-250', '360-287-8700');
       INSERT INTO DEPARTMENT VALUES(
'Production', 'BC-900-10', 'BLDG02-110', '360-287-8800');

/***** EMPLOYEE DATA *****************************************************/

INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, OfficePhone, EmailAddress)
       VALUES('Mary', 'Jacobs', 'Administration', 'CEO',
       '360-285-8110', 'Mary.Jacobs@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
       VALUES('Rosalie', 'Jackson', 'Administration', 'Admin Assistant', 1,
       '360-285-8120', 'Rosalie.Jackson@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
       VALUES('Richard', 'Bandalone', 'Legal', 'Attorney', 1,
       '360-285-8210', 'Richard.Bandalone@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
       VALUES('George', 'Smith', 'Human Resources', 'HR3', 1,
       '360-285-8310', 'George.Smith@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
       VALUES('Alan', 'Adams', 'Human Resources', 'HR1', 4,
       '360-285-8320', 'Alan.Adams@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
       VALUES('Ken', 'Evans', 'Finance', 'CFO', 1,
       '360-285-8410', 'Ken.Evans@WP.com');
```

Figure C-19 — The WP Database Data INSERT Statements for MySQL

```
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Mary', 'Abernathy', 'Finance', 'FA3', 6,
      '360-285-8420', 'Mary.Abernathy@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES(
      'Tom', 'Caruthers', 'Accounting', 'FA2', 6,
      '360-285-8430', 'Tom.Caruthers@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Heather', 'Jones', 'Accounting', 'FA2', 6,
      '360-285-8440', 'Heather.Jones@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Ken', 'Numoto', 'Sales and Marketing', 'SM3', 1,
      '360-287-8510', 'Ken.Numoto@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Linda', 'Granger', 'Sales and Marketing', 'SM2', 10,
      '360-287-8520', 'Linda.Granger@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('James', 'Nestor', 'InfoSystems', 'CIO', 1,
      '360-287-8610', 'James.Nestor@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, EmailAddress)
      VALUES('Rick', 'Brown', 'InfoSystems', 'IS2', 12, 'Rick.Brown@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Mike', 'Nguyen', 'Research and Development', 'CTO', 1,
      '360-287-8710', 'Mike.Nguyen@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Jason', 'Sleeman', 'Research and Development', 'RD3', 14,
      '360-287-8720', 'Jason.Sleeman@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Mary', 'Smith', 'Production', 'OPS3', 1,
      '360-287-8810', 'Mary.Smith@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('Tom', 'Jackson', 'Production', 'OPS2', 14,
      '360-287-8820', 'Tom.Jackson@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, OfficePhone, EmailAddress)
      VALUES('George', 'Jones', 'Production', 'OPS2', 15,
      '360-287-8830', 'George.Jones@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, EmailAddress)
      VALUES('Julia', 'Hayakawa', 'Production', 'OPS1', 15, 'Julia.Hayakawa@WP.com');
INSERT INTO EMPLOYEE
(FirstName, LastName, Department, Position, Supervisor, EmailAddress)
      VALUES('Sam', 'Stewart', 'Production', 'OPS1', 15, 'Sam.Stewart@WP.com');
```

Figure C-19 — The WP Database Data INSERT Statements for MySQL (continued)

```
/***** PROJECT DATA ************************************************************/

INSERT INTO PROJECT VALUES(
       1000,'2017 Q3 Production Plan', 'Production', 100.00, '2017-05-10',
       '2017-06-15');
INSERT INTO PROJECT VALUES(
       1100,'2017 Q3 Marketing Plan', 'Sales and Marketing', 135.00, '2017-05-10',
       '2017-06-15');
INSERT INTO PROJECT VALUES(
       1200,'2017 Q3 Portfolio Analysis', 'Finance', 120.00, '2017-07-05',
       '2017-07-25');
INSERT INTO PROJECT VALUES(
       1300,'2017 Q3 Tax Preparation', 'Accounting', 145.00, '2017-08-10',
        '2017-10-15');
INSERT INTO PROJECT VALUES(
       1400,'2017 Q4 Production Plan', 'Production', 100.00, '2017-08-10',
        '2017-09-15');
INSERT INTO PROJECT VALUES(
       1500,'2017 Q4 Marketing Plan', 'Sales and Marketing', 135.00, '2017-08-10',
        '2017-09-15');
INSERT INTO PROJECT(ProjectID, ProjectName, Department, MaxHours, StartDate)
       VALUES(1600,'2017 Q4 Portfolio Analysis', 'Finance', 140.00, '2017-10-05');


/***** ASSIGNMENT DATA *********************************************************/

INSERT INTO ASSIGNMENT VALUES(1000, 1, 30.0);
INSERT INTO ASSIGNMENT VALUES(1000, 6, 50.0);
INSERT INTO ASSIGNMENT VALUES(1000, 10, 50.0);
INSERT INTO ASSIGNMENT VALUES(1000, 16, 75.0);
INSERT INTO ASSIGNMENT VALUES(1000, 17, 75.0);
INSERT INTO ASSIGNMENT VALUES(1100, 1, 30.0);
INSERT INTO ASSIGNMENT VALUES(1100, 6, 75.0);
INSERT INTO ASSIGNMENT VALUES(1100, 10, 55.0);
INSERT INTO ASSIGNMENT VALUES(1100, 11, 55.0);
INSERT INTO ASSIGNMENT VALUES(1200, 3, 20.0);
INSERT INTO ASSIGNMENT VALUES(1200, 6, 40.0);
INSERT INTO ASSIGNMENT VALUES(1200, 7, 45.0);
INSERT INTO ASSIGNMENT VALUES(1200, 8, 45.0);
INSERT INTO ASSIGNMENT VALUES(1300, 3, 25.0);
INSERT INTO ASSIGNMENT VALUES(1300, 6, 40.0);
INSERT INTO ASSIGNMENT VALUES(1300, 8, 50.0);
INSERT INTO ASSIGNMENT VALUES(1300, 9, 50.0);
INSERT INTO ASSIGNMENT VALUES(1400, 1, 30.0);
INSERT INTO ASSIGNMENT VALUES(1400, 6, 50.0);
INSERT INTO ASSIGNMENT VALUES(1400, 10, 50.0);
INSERT INTO ASSIGNMENT VALUES(1400, 16, 75.0);
INSERT INTO ASSIGNMENT VALUES(1400, 17, 75.0);
INSERT INTO ASSIGNMENT VALUES(1500, 1, 30.0);
INSERT INTO ASSIGNMENT VALUES(1500, 6, 75.0);
INSERT INTO ASSIGNMENT VALUES(1500, 10, 55.0);
INSERT INTO ASSIGNMENT VALUES(1500, 11, 55.0);
INSERT INTO ASSIGNMENT VALUES(1600, 3, 20.0);
INSERT INTO ASSIGNMENT VALUES(1600, 6, 40.0);
INSERT INTO ASSIGNMENT VALUES(1600, 7, 45.0);
INSERT INTO ASSIGNMENT VALUES(1600, 8, 45.0);

/*******************************************************************************/
```

Figure C-19 — The WP Database Data INSERT Statements for MySQL (continued)

How Do I Work with SQL Queries in MySQL?

Now that we've created and populated the WP database, we can run SQL queries against the data. While scripts are good for large sets of SQL commands that need to be run together, most SQL queries are run as single commands. Nonetheless, individual SQL queries are created in the same tabbed window and run the same way as SQL scripts. Use the **Create a new SQL tab for executing queries** button as shown in Figure C-19 to open a new query window, enter a simple SQL query, and execute it.

Figure C-20 shows an SQL query with its results. SQL queries can also be named and saved. The results of the SQL query appear in a tabbed results window in a spreadsheet-style display. You can modify the column widths in the results display by using standard Windows drag-and-drop techniques to help make more data visible. You can run multiple queries at the same time by using multiple results windows. You can open additional SQL Script tabbed windows so that you can work with multiple SQL queries or SQL scripts at the same time. You can also highlight a single SQL command in a script and run it separately using the **Execute the statement under the keyboard cursor** button shown in Figure C-20.

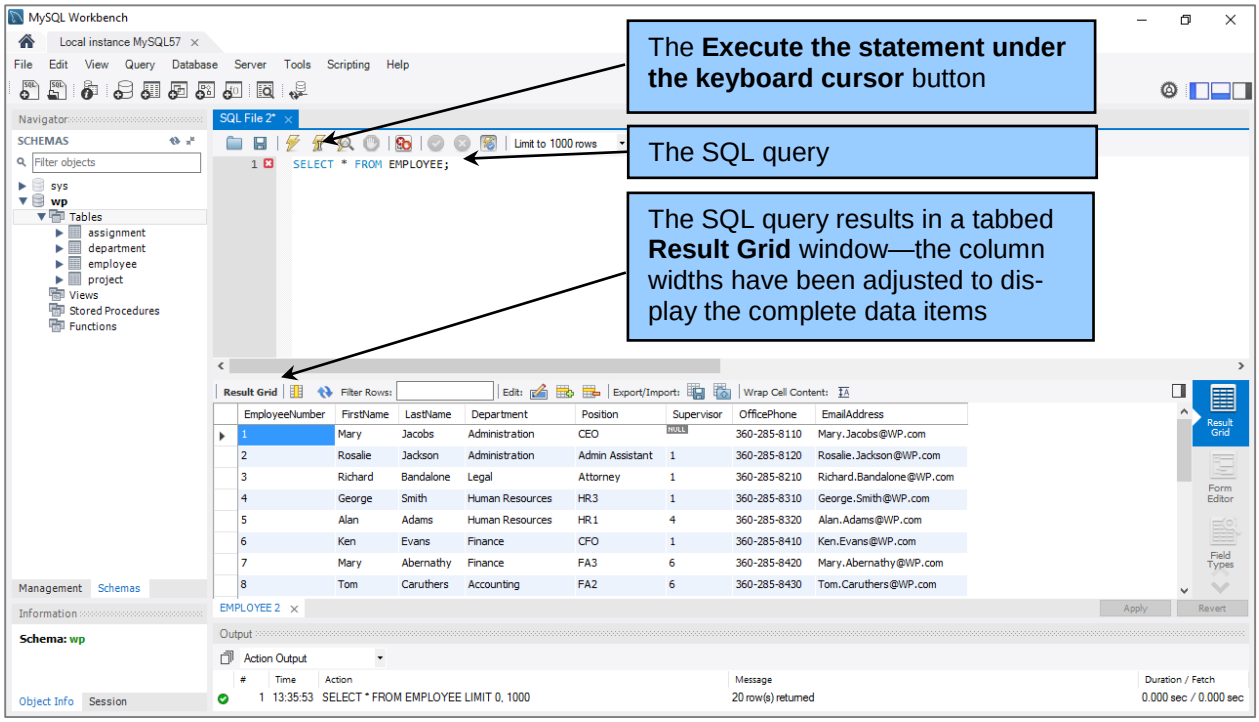


Figure C-20 — The SQL Query and Results in the MySQL Workbench

By The Way

The folks who work on the MySQL Workbench have **no qualms** about making **major GUI changes** in minor version upgrades of the product. The same functionality will be there, but perhaps implemented slightly differently! **Be aware and be prepared!**

Additional Documentation for MySQL

To get access to MySQL documentation, use the **Help | Help Index** command for help on MySQL Workbench itself, and go to http://dev.mysql.com/doc for documentation on MySQL 5.7 (and other versions of MySQL).

By The Way

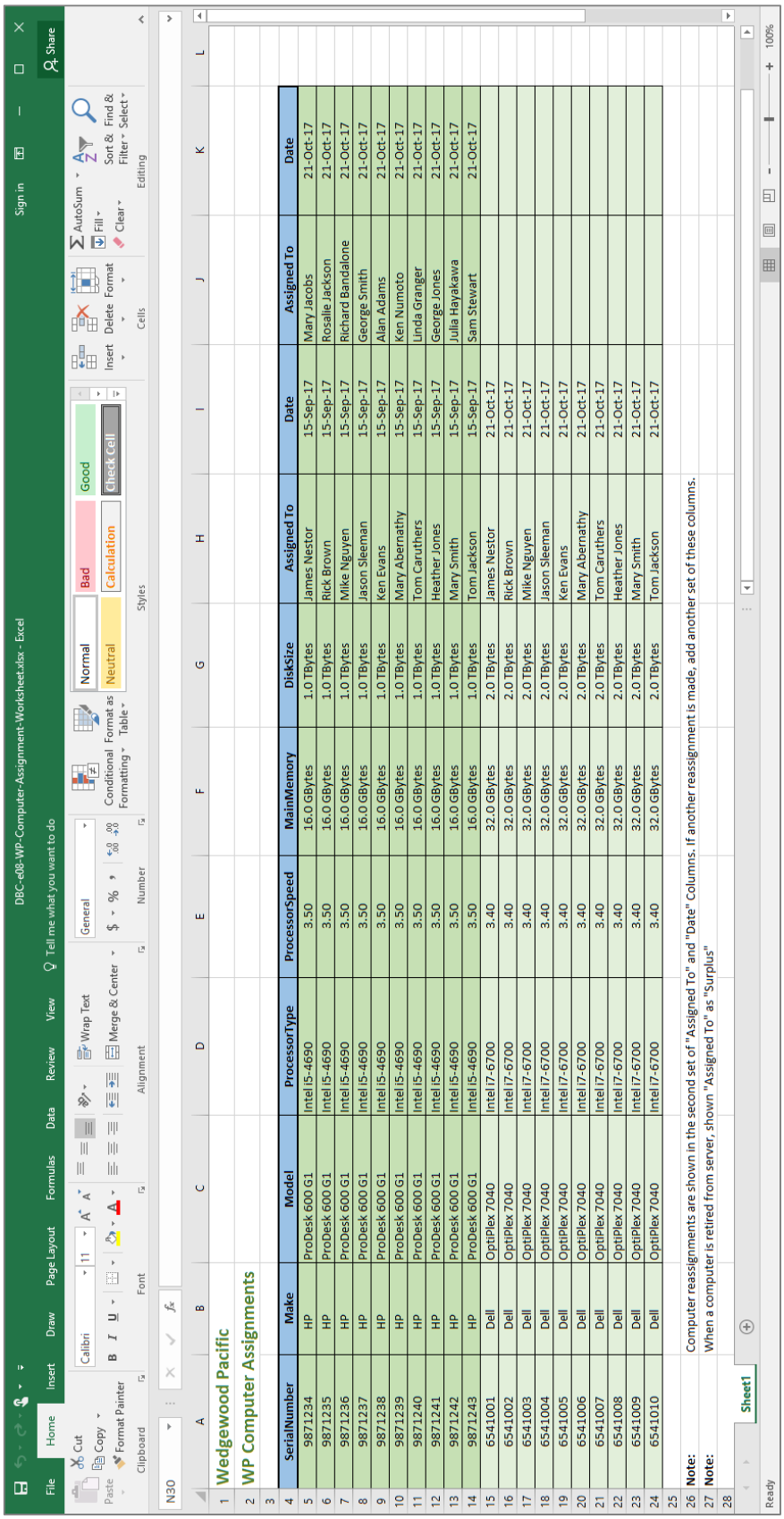
Before proceeding with the rest of the material in this Appendix, we recommend that you work through and understand the SQL topics covered in Chapter 3, "Structured Query Language," and Appendix E, "Advanced SQL."

How Do I Import Microsoft Excel Data into a MySQL 5.7 Table?

When developing a database to support an application, it is very common to find that some (if not all) of the data needed in the database exists as data in user **worksheets** (also called **spreadsheets**). A typical example of this is a Microsoft Excel 2016 worksheet that a user has been maintaining and must now be converted to data stored in the database. If we are really lucky, the worksheet will already be organized like a database table, with appropriate column labels and unique data in each row. And if we are *really, really lucky*, there will be one or more columns that can be used as the primary key in the new database table. In that case, we can easily import the data into the database. More likely, we will have to modify the worksheet and organize and clean up the data in it before we can import the data. In essence, we are following a procedure that we encounter in Chapter 8 in our discussion of data warehouses known as **extract, transform, and load (ETL)**.

As an example, let's consider the problem of computers owned by WP. WP needs to track these computers (asset inventory) and who they are currently and been assigned to for use. The properly designed tables (COMPUTER and COMPUTER_ASSIGNMENT) to handle this problem are shown in the Chapter 3 Access Workbench Exercises as Figures 3-32 and 3-34. The data for the tables is shown in Figures 3-33 and 3-35.

Unfortunately, that is not the way we will probably encounter the data. More likely we'll find it stored in a worksheet such as the Microsoft Excel 2016 worksheet shown in Figure C-21. This worksheet breaks our basic rule of one theme per table and combines computer inventory and computer assignment data into the same worksheet. Worse, the computer assignments are handled by using multiple assignment and date columns. This is an example of what is called the **multivalue, multicolumn problem**, which occurs when multiple columns are used in a spreadsheet or database table to record repetitions of the same data. A good example is EMPLOYEE phone number data, where we might find columns for



Wedgewood Pacific

WP Computer Assignments

SerialNumber	Make	Model	ProcessorType	ProcessorSpeed	MainMemory	DiskSize	Assigned To	Date	Assigned To	Date
9871234	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	James Nestor	15-Sep-17	Mary Jacobs	21-Oct-17
9871235	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Rick Brown	15-Sep-17	Rosalie Jackson	21-Oct-17
9871236	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Mike Nguyen	15-Sep-17	Richard Bandalone	21-Oct-17
9871237	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Jason Sleeman	15-Sep-17	George Smith	21-Oct-17
9871238	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Ken Evans	15-Sep-17	Alan Adams	21-Oct-17
9871239	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Mary Abernathy	15-Sep-17	Ken Numoto	21-Oct-17
9871240	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Tom Caruthers	15-Sep-17	Linda Granger	21-Oct-17
9871241	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Heather Jones	15-Sep-17	George Jones	21-Oct-17
9871242	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Mary Smith	15-Sep-17	Julia Hayakawa	21-Oct-17
9871243	HP	ProDesk 600 G1	Intel I5-4690	3.50	16.0 GBytes	1.0 TBytes	Tom Jackson	15-Sep-17	Sam Stewart	21-Oct-17
6541001	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	James Nestor	21-Oct-17		
6541002	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Rick Brown	21-Oct-17		
6541003	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Mike Nguyen	21-Oct-17		
6541004	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Jason Sleeman	21-Oct-17		
6541005	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Ken Evans	21-Oct-17		
6541006	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Mary Abernathy	21-Oct-17		
6541007	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Tom Caruthers	21-Oct-17		
6541008	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Heather Jones	21-Oct-17		
6541009	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Mary Smith	21-Oct-17		
6541010	Dell	OptiPlex 7040	Intel I7-6700	3.40	32.0 GBytes	2.0 TBytes	Tom Jackson	21-Oct-17		

Note: Computer reassignments are shown in the second set of "Assigned To" and "Date" Columns. If another reassignment is made, add another set of these columns.

Note: When a computer is retired from server, shown "Assigned To" as "Surplus"

Figure C-21 — The WP Computer Assignments in a Microsoft Excel 2016 Worksheet

HomePhone, CellPhone, and BusinessPhone. This may seem reasonable until we have to add yet *another* phone number, perhaps DepartmentPhone or SpousesPhone. What we are dealing with here is a **multivalued dependency** (as discussed in Chapter 2), where the determinant determines multiple values instead of just one:

EmployeeID $\rightarrow\!\!\rightarrow$ PhoneNumber

A detailed solution to this problem is beyond the scope of this book, but the basic answer to use 4NF and put this dependency into its own table.[7]

In our current situation, it is obvious that we must *extract* the data we need from the worksheet for two database tables (COMPUTER and COMPUTER_ASSIGNMENT), *transform* each set of data into a correctly structured and formatted worksheet, and then *load* (import) the data from the worksheet into the database.

We can do this by:

- Creating two new worksheets named COMPUTER and COMPUTER_ASSIGNMENT and copying the data into them, then
- Modify the structure and data in each worksheet so that it is correct for importing into the database, then
- Import the data from each worksheet into the database

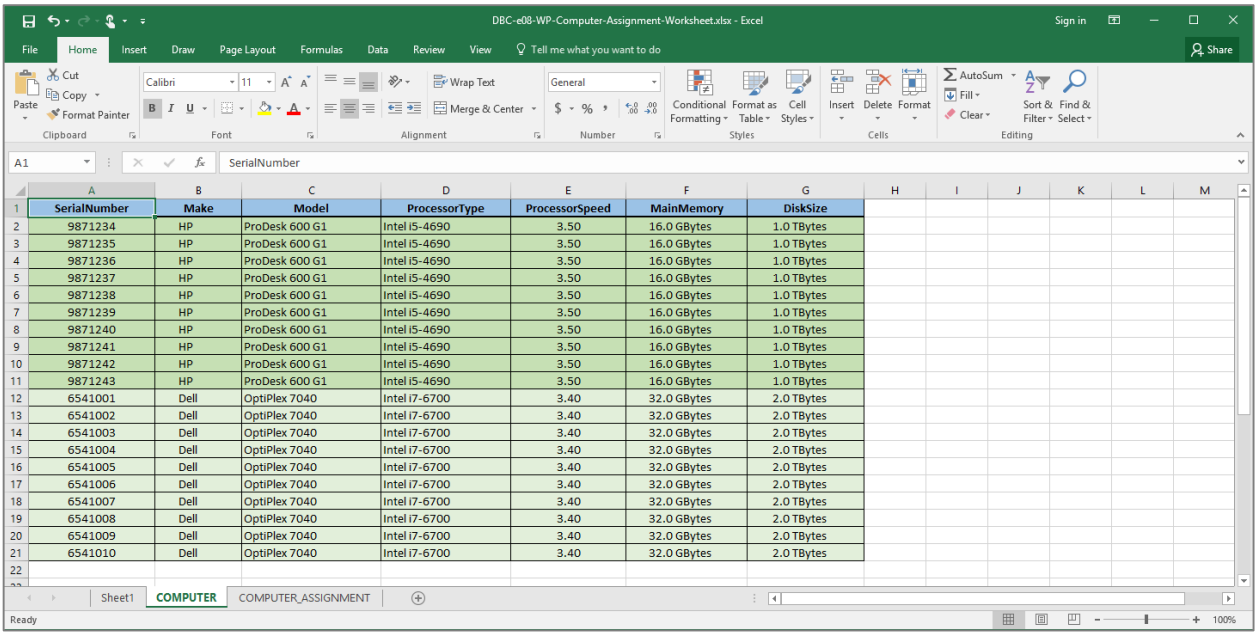
After the data is imported into two database tables, we will have to use SQL ALTER TABLE statements to create primary keys, foreign keys, and any other needed constraints—while we will use the SQL ALTER TABLE statement as needed here, a full discussion of the SQL ALTER TABLE statement can be found in Appendix E, "Advanced SQL."

Preparing the Microsoft Excel Data for Import into a Database Table

Figure C-22 shows the COMPUTER worksheet after it has been cleaned up. All extraneous rows and columns have been deleted, and only the computer data (with appropriate columns headers) remains. This worksheet now looks like a database table, which is a good indication that the data import should work properly.

Figure C-23 shows the COMPUTER_ASSIGNMENT worksheet after our first attempt at restructuring it. There are still some obvious problems here. First of all, in the WP database we identify employees by their *EmployeeNumber*, not by their name. Second, we still have multiple Assigned To and Date columns. Therefore, we need to (1) substitute *EmployeeNumber* for *Assigned To* and (2) combine the Assigned to and Date columns.

[7] For more information about multivalued dependencies and the multivalue, multicolumn problem, see David Kroenke and David Auer, *Database Processing: Fundamentals, Design, and Implementation* (14th edition) (Upper Saddle River, NJ: Pearson Higher Education, 2016).

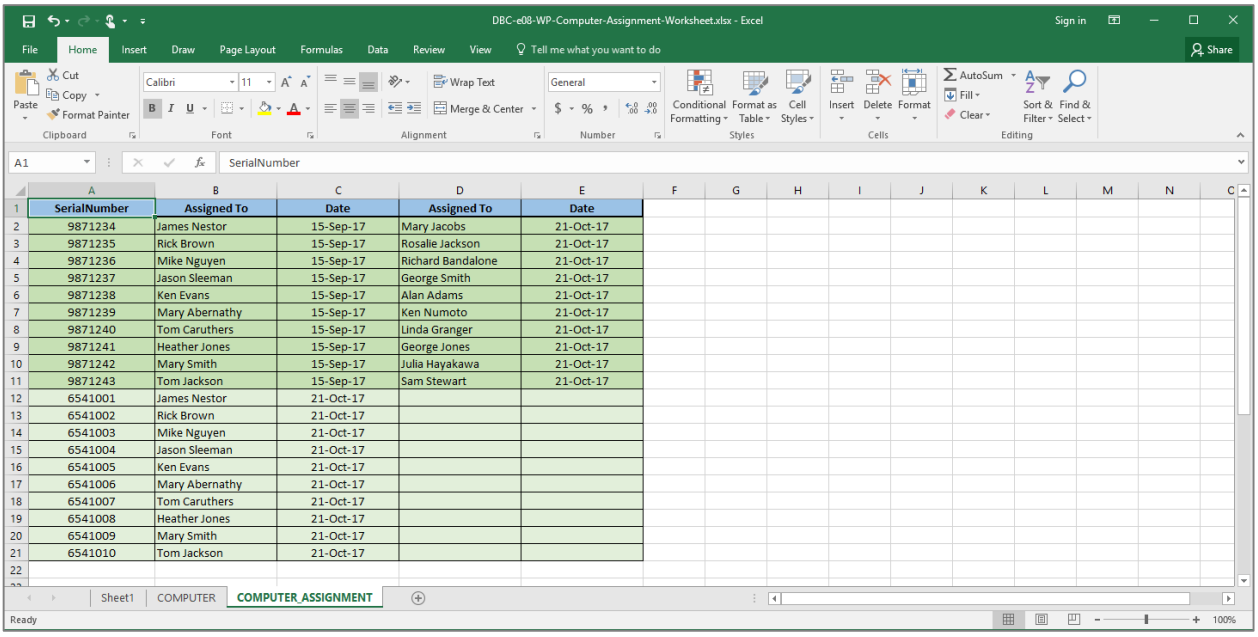


DBC-e08-WP-Computer-Assignment-Worksheet.xlsx - Excel

SerialNumber	Make	Model	ProcessorType	ProcessorSpeed	MainMemory	DiskSize
9871234	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871235	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871236	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871237	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871238	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871239	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871240	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871241	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871242	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
9871243	HP	ProDesk 600 G1	Intel i5-4690	3.50	16.0 GBytes	1.0 TBytes
6541001	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541002	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541003	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541004	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541005	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541006	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541007	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541008	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541009	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes
6541010	Dell	OptiPlex 7040	Intel i7-6700	3.40	32.0 GBytes	2.0 TBytes

Sheet1 | COMPUTER | COMPUTER_ASSIGNMENT

Figure C-22 — The WP COMPUTER Worksheet



DBC-e08-WP-Computer-Assignment-Worksheet.xlsx - Excel

SerialNumber	Assigned To	Date	Assigned To	Date
9871234	James Nestor	15-Sep-17	Mary Jacobs	21-Oct-17
9871235	Rick Brown	15-Sep-17	Rosalie Jackson	21-Oct-17
9871236	Mike Nguyen	15-Sep-17	Richard Bandalone	21-Oct-17
9871237	Jason Sleeman	15-Sep-17	George Smith	21-Oct-17
9871238	Ken Evans	15-Sep-17	Alan Adams	21-Oct-17
9871239	Mary Abernathy	15-Sep-17	Ken Numoto	21-Oct-17
9871240	Tom Caruthers	15-Sep-17	Linda Granger	21-Oct-17
9871241	Heather Jones	15-Sep-17	George Jones	21-Oct-17
9871242	Mary Smith	15-Sep-17	Julia Hayakawa	21-Oct-17
9871243	Tom Jackson	15-Sep-17	Sam Stewart	21-Oct-17
6541001	James Nestor	21-Oct-17		
6541002	Rick Brown	21-Oct-17		
6541003	Mike Nguyen	21-Oct-17		
6541004	Jason Sleeman	21-Oct-17		
6541005	Ken Evans	21-Oct-17		
6541006	Mary Abernathy	21-Oct-17		
6541007	Tom Caruthers	21-Oct-17		
6541008	Heather Jones	21-Oct-17		
6541009	Mary Smith	21-Oct-17		
6541010	Tom Jackson	21-Oct-17		

Sheet1 | COMPUTER | COMPUTER_ASSIGNMENT

Figure C-23 — The WP COMPUTER_ASSIGNMENT Worksheet – First Attempt

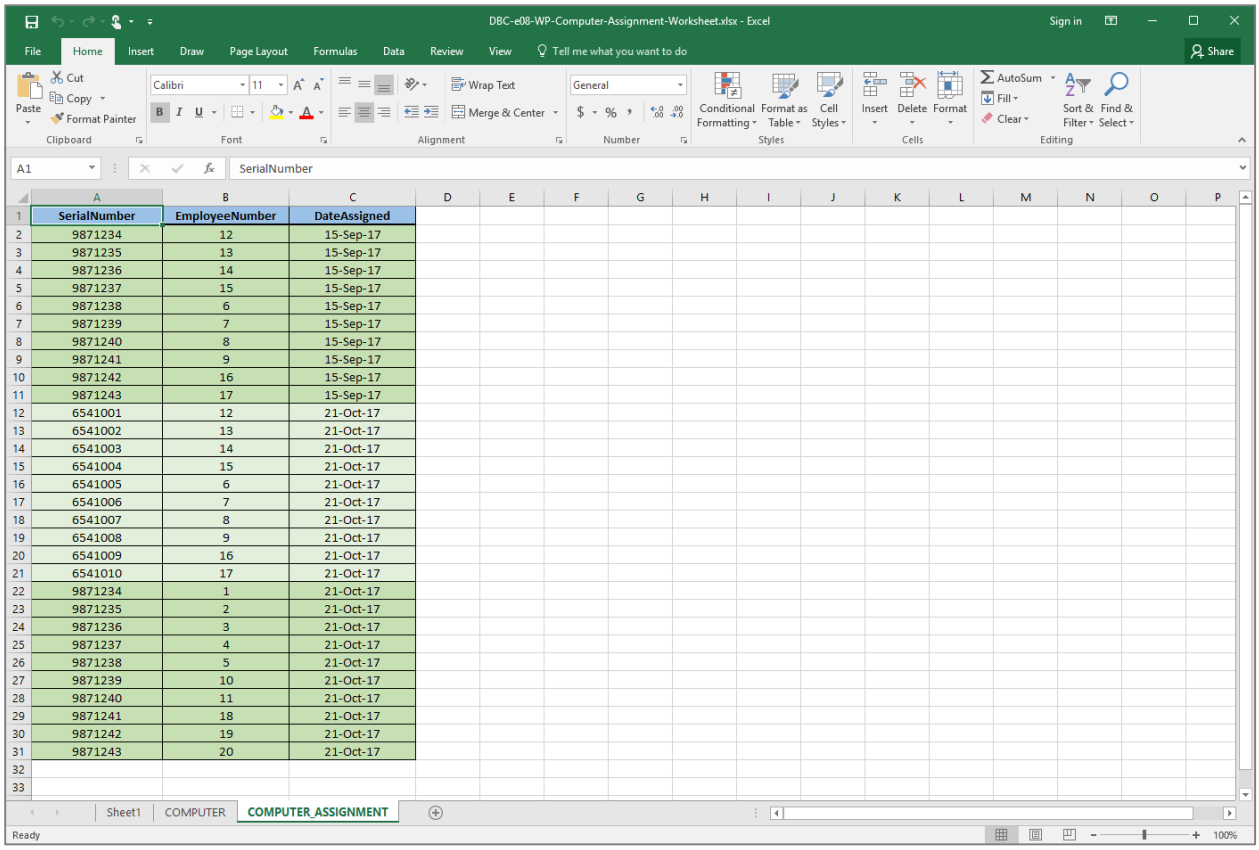
We can determine EmployeeNumber (a surrogate key) by using an SQL query in the WP database:

```
/* *** SQL-Query-AppC-01 *** */

SELECT   *   FROM   EMPLOYEE;
```

This query gives us:

EmployeeNumber	FirstName	LastName	Department	Position	Supervisor	OfficePhone	EmailAddress
1	Mary	Jacobs	Administration	CEO	NULL	360-285-8110	Mary.Jacobs@WP.com
2	Rosalie	Jackson	Administration	Admin Assistant	1	360-285-8120	Rosalie.Jackson@WP.com
3	Richard	Bandalone	Legal	Attorney	1	360-285-8210	Richard.Bandalone@WP.com
4	George	Smith	Human Resources	HR3	1	360-285-8310	George.Smith@WP.com
5	Alan	Adams	Human Resources	HR1	4	360-285-8320	Alan.Adams@WP.com
6	Ken	Evans	Finance	CFO	1	360-285-8410	Ken.Evans@WP.com
7	Mary	Abernathy	Finance	FA3	6	360-285-8420	Mary.Abernathy@WP.com
8	Tom	Caruthers	Accounting	FA2	6	360-285-8430	Tom.Caruthers@WP.com
9	Heather	Jones	Accounting	FA2	6	360-285-8440	Heather.Jones@WP.com
10	Ken	Numoto	Sales and Marketing	SM3	1	360-287-8510	Ken.Numoto@WP.com
11	Linda	Granger	Sales and Marketing	SM2	10	360-287-8520	Linda.Granger@WP.com
12	James	Nestor	InfoSystems	CIO	1	360-287-8610	James.Nestor@WP.com
13	Rick	Brown	InfoSystems	IS2	12	NULL	Rick.Brown@WP.com
14	Mike	Nguyen	Research and Development	CTO	1	360-287-8710	Mike.Nguyen@WP.com
15	Jason	Sleeman	Research and Development	RD3	14	360-287-8720	Jason.Sleeman@WP.com
16	Mary	Smith	Production	OPS3	1	360-287-8810	Mary.Smith@WP.com
17	Tom	Jackson	Production	OPS2	14	360-287-8820	Tom.Jackson@WP.com
18	George	Jones	Production	OPS2	15	360-287-8830	George.Jones@WP.com
19	Julia	Hayakawa	Production	OPS1	15	NULL	Julia.Hayakawa@WP.com
20	Sam	Stewart	Production	OPS1	15	NULL	Sam.Stewart@WP.com
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



SerialNumber	EmployeeNumber	DateAssigned
9871234	12	15-Sep-17
9871235	13	15-Sep-17
9871236	14	15-Sep-17
9871237	15	15-Sep-17
9871238	6	15-Sep-17
9871239	7	15-Sep-17
9871240	8	15-Sep-17
9871241	9	15-Sep-17
9871242	16	15-Sep-17
9871243	17	15-Sep-17
6541001	12	21-Oct-17
6541002	13	21-Oct-17
6541003	14	21-Oct-17
6541004	15	21-Oct-17
6541005	6	21-Oct-17
6541006	7	21-Oct-17
6541007	8	21-Oct-17
6541008	9	21-Oct-17
6541009	16	21-Oct-17
6541010	17	21-Oct-17
9871234	1	21-Oct-17
9871235	2	21-Oct-17
9871236	3	21-Oct-17
9871237	4	21-Oct-17
9871238	5	21-Oct-17
9871239	10	21-Oct-17
9871240	11	21-Oct-17
9871241	18	21-Oct-17
9871242	19	21-Oct-17
9871243	20	21-Oct-17

Figure C-24 — The WP COMPUTER_ASSIGNMENT Worksheet – Second Attempt

Database Column Characteristics for the WP COMPUTER Table

Column Name	Type	Key	Required	Remarks
SerialNumber	Number	Primary Key	Yes	Long Integer
Make	Short Text (12)	No	Yes	Must be "Dell" or "Gateway" or "HP" or "Other"
Model	Short Text (24)	No	Yes	
ProcessorType	Short Text (24)	No	No	
ProcessorSpeed	Number	No	Yes	Double [3,2], Between 1.0 and 4.0
MainMemory	Short Text (15)	No	Yes	
DiskSize	Short Text (15)	No	Yes	

Figure C-25 — Database Column Characteristics for the WP COMPUTER Table (from Figure 3-32)

Using this data, we can rework the COMPUTER_ASSIGNMENT worksheet as shown in Figure C-24, which also includes a renamed *Date* column which is now *DateAssigned.* Admittedly this is a small example, and given a larger data set a different strategy would be needed. For our purposes here, however, this method will work. We will now look at how to import a table into MySQL 5.7. The COMPUTER table column characteristics as stated in Figure 3-32 are shown below in Figure C-25. We would normally have to add CHECK constraints on this table after the data import, but unfortunately MySQL does not support CHECK constraints.

Importing the Microsoft Excel Data into a MySQL 5.7 Database Table

For MySQL, we will create the COMPUTER Table using the **MySQL for Excel Add-In**. We installed this utility using the MySQL Installer, and when Microsoft Excel is launched, it will then appear on the DATA tab in the Microsoft Excel 2016 ribbon. The MySQL for Excel Add-In does a good job of letting us create a new table, set a primary key and specify most column characteristics. It does not, however, let us set CHECK CONSTRAINT constraints as specified in Figure C-25, so we will have to use SQL ALTER TABLE statements to add those. However, MySQL does not support some common SQL ALTER TABLE features, so we will have to use MySQL specific syntax. See http://dev.mysql.com/doc/refman/5.76/en/alter-table.html.

1. Open the DBC-e08-WP-Computer-Assignment-Worksheet in Microsoft Excel 2016, click the DATA tab in the Ribbon. The MySQL for Excel button is displayed, as shown in Figure C-26. Click the **MySQL for Excel** button to launch the MySQL for Excel pane, as shown in Figure C-27.
2. Open a MySQL connection by double-clicking **Local instance MySQL57** and logging into the MySQL 5.7 server.
3. As shown in Figure C-28, click the wp schema name to select it, and then click the **Next** button.
4. In Microsoft Excel, select (highlight) the entire COMPUTER table range!
5. As shown in Figure C-29, click the **Export Excel Data to New Table** command. The Export Data dialog box is displayed, as shown in figure C-30, labeled with the name of the selected Microsoft Excel sheet and the selected range (COMPUTER [A1:G21]).
6. Complete the new COMPUTER table specifications to as shown below to approximate Figure 3-32 – note that you can adjust data types and NULL/NOT NULL (shown as "Allow Empty") for each column as shown in Figure C-31. Although Figure 3-32 shows Text data types

which would normally be CHAR data types, we will use the selected MySQL VARCHAR data type for text columns and adjust the number of characters to match the following characteristics, as MySQL complains about some of the data sizes in Figure 3-32. Click the column heading for each and adjust the attribute characteristics as follows:

SerialNumber	Integer – Primary Key
Make	VarChar(12)
Model	VarChar(24)
ProcessorType	VarChar(24) – Allow Empty (NULL)
ProcessorSpeed	Decimal(12,2)
MainMemory	VarChar(25)
DiskSize	VarChar(25)

7. The complete *Export Data – COMPUTER [A1:G21]* dialog box is shown in Figure C-31.
8. Click the **Export Data** button. The new table is created and populated, as shown in the Success dialog box seen in Figure C-32.
9. In the Success dialog box, click the **OK** button.
10. In the Microsoft Excel *MySQL For Excel* pane, click the **Close** button to close MySQL for Excel.
11. Save the Microsoft Excel workbook. If a dialog box appears warning about macro features that cannot be saved, ignore it and click the **Yes** button.
12. Close the Microsoft Excel workbook.

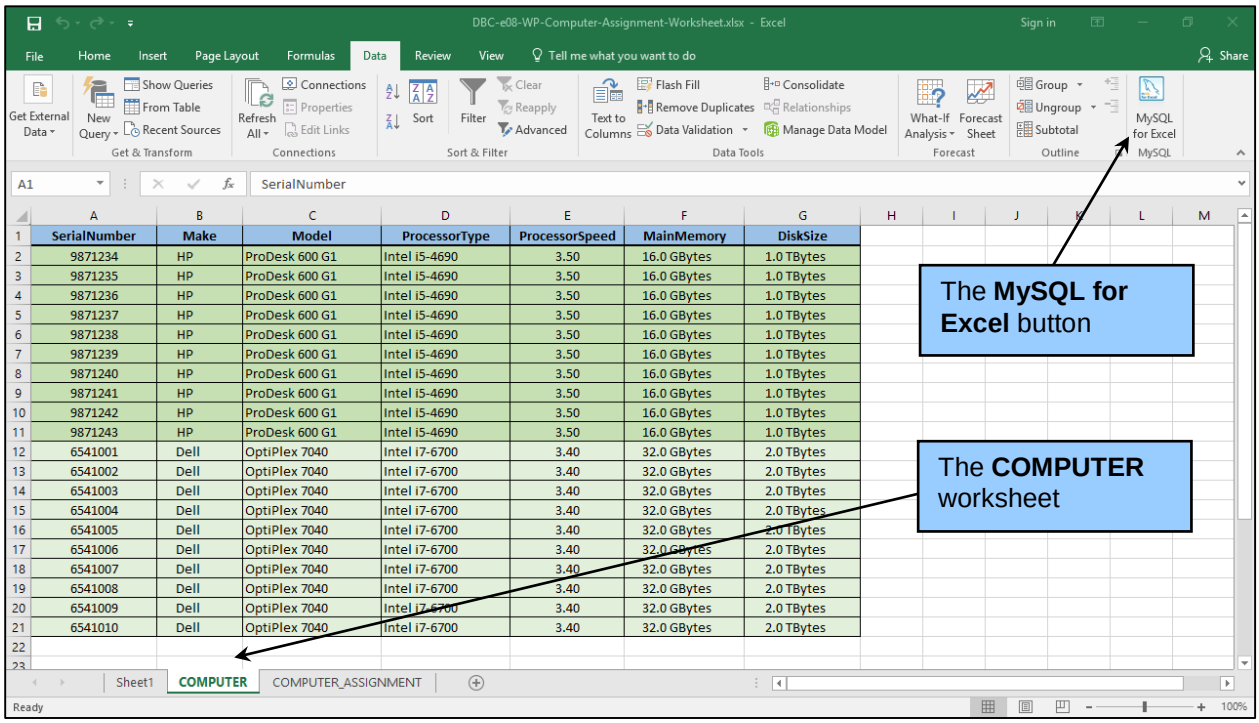


Figure C-26 — The MySQL for Excel Button

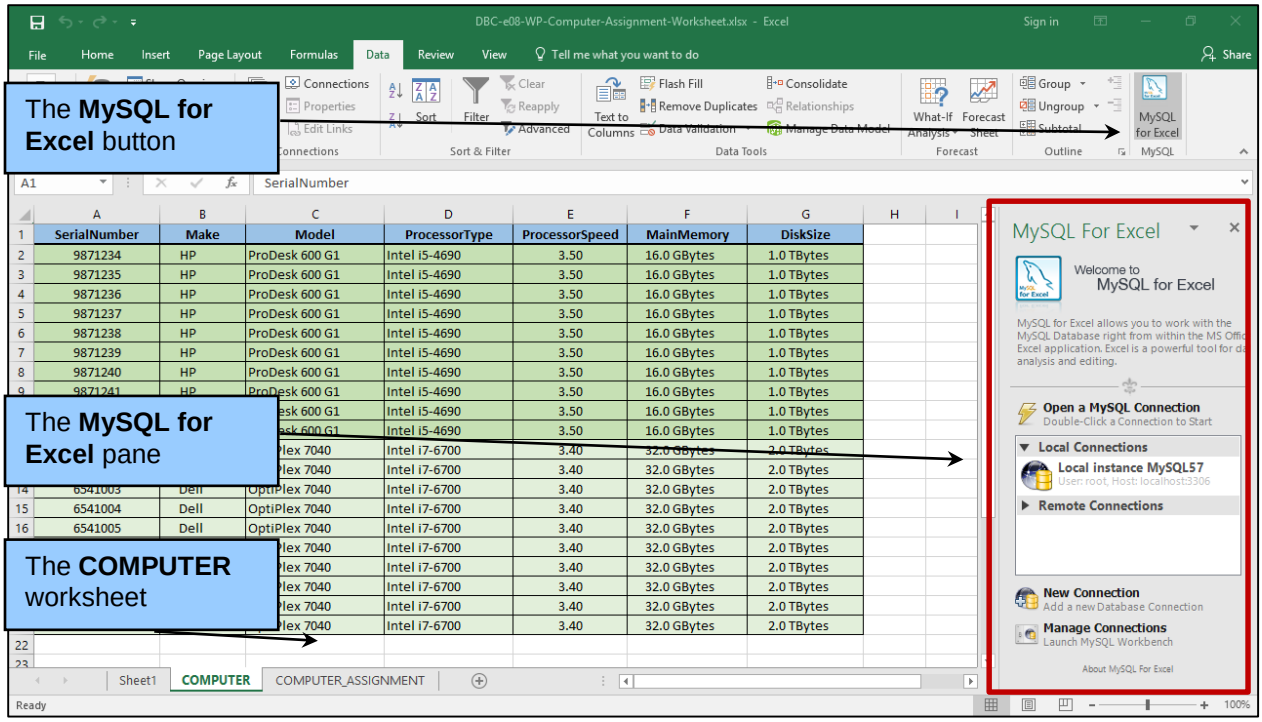


Figure C-27 — The MySQL for Excel Pane

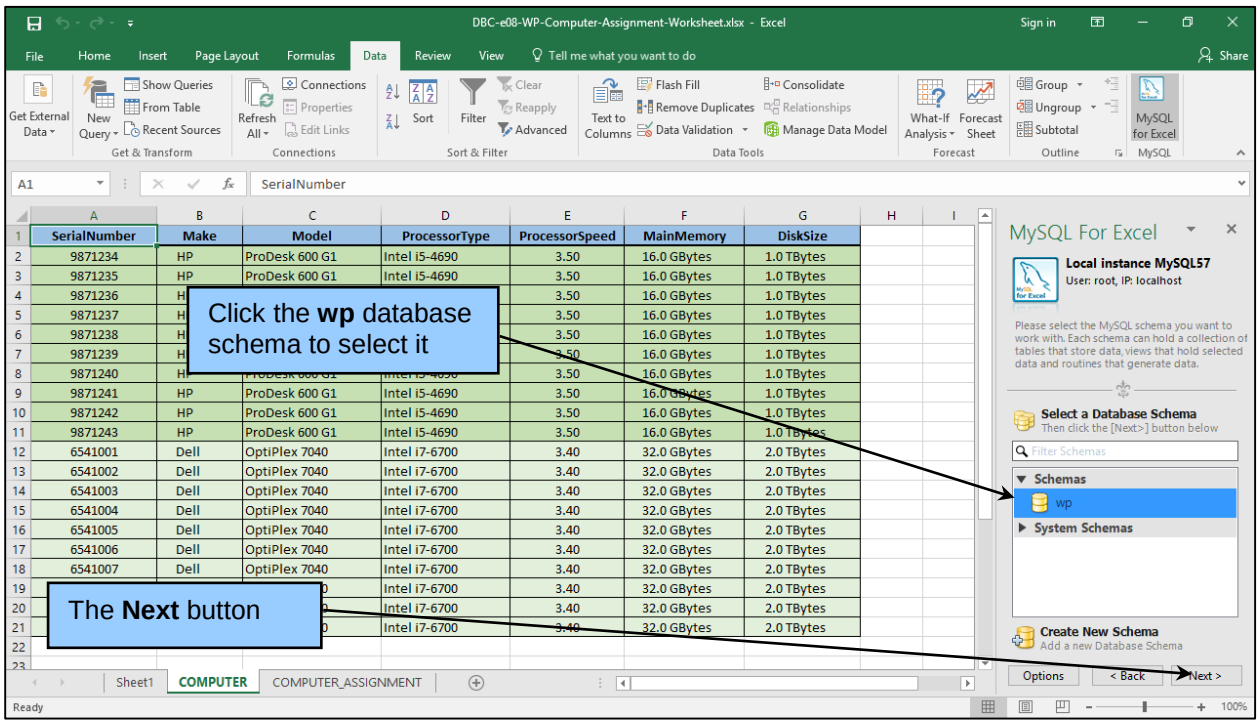


Figure C-28 — Selecting the wp Database Schema

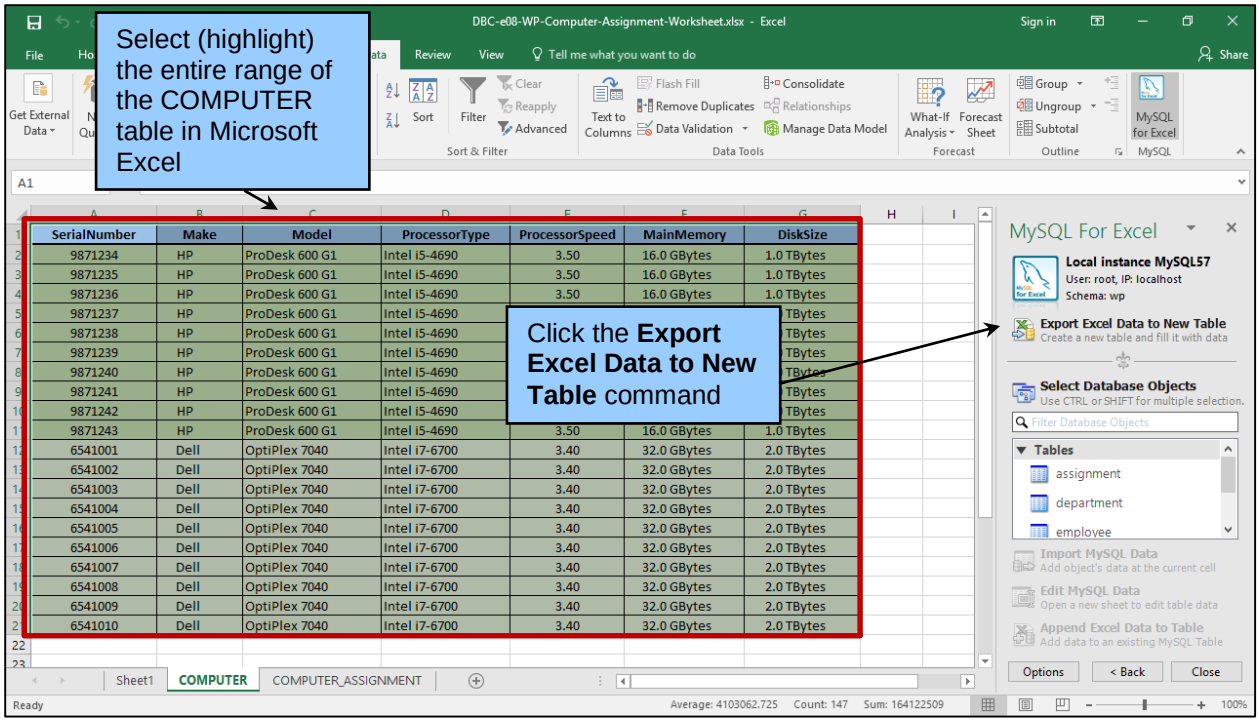


Figure C-29 — The MySQL for Excel Pane

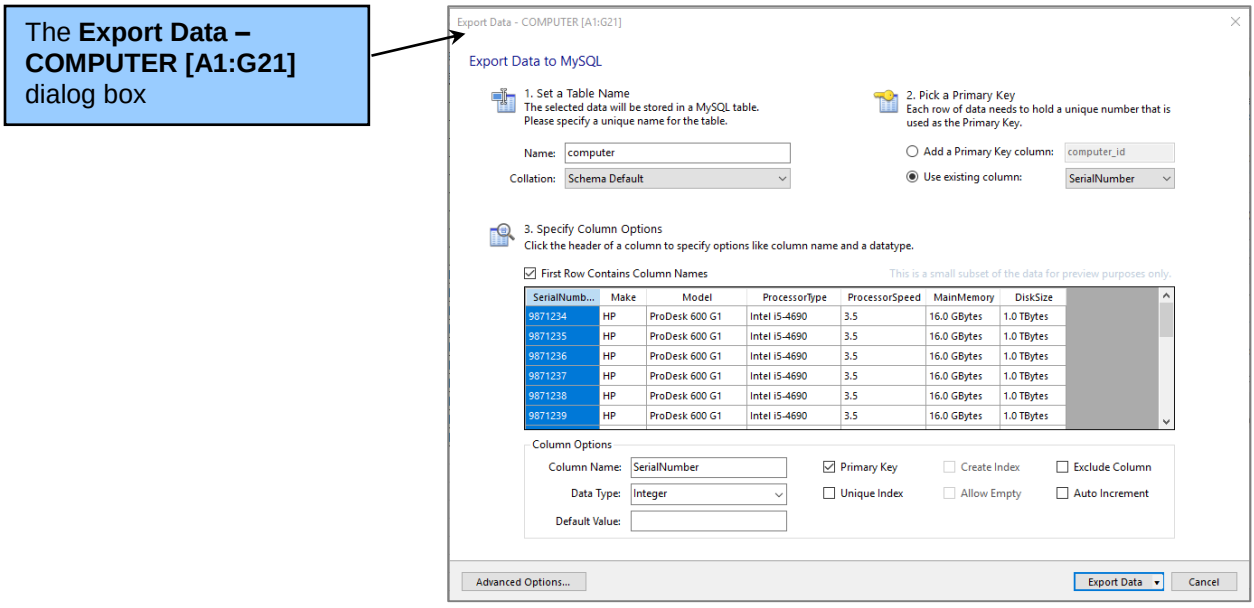


Figure C-30 — The Export Data – COMPUTER [A1:G21] Dialog Box

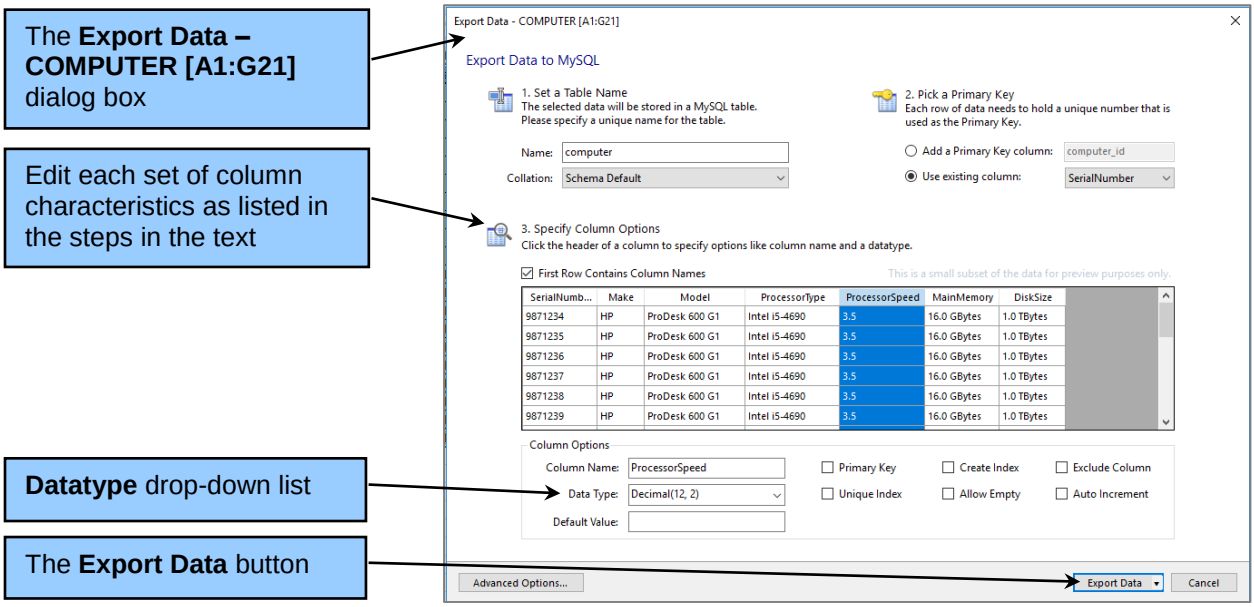


Figure C-31 — Editing the COMPUTER Table Specifications

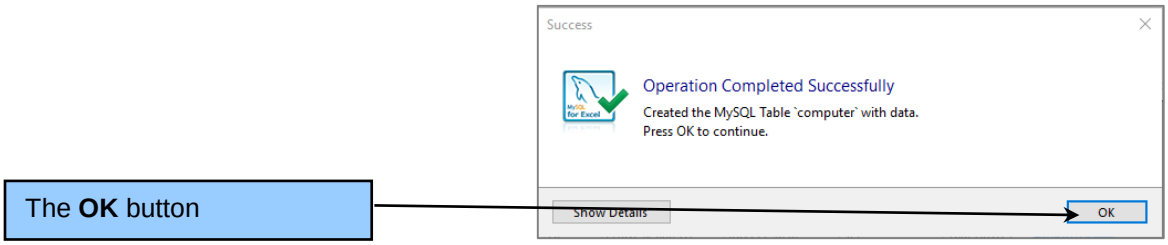


Figure C-32 — The Success Dialog Box

13. Open MySQL Workbench, login if needed, and refresh the **wp** schema. The "refresh" button is shown in Figure C-20, and once refreshed, the SCHEMAS section will show the new COMPUTER table. Expand the Tables object and the *computer* table object Columns.

14. We need to inspect the structure of the new *computer* table. Right-click the **computer table object**, click **Table Inspector**, and then click the **Columns** tab. The correct column characteristics for the *computer* table are displayed as shown in Figure C-33. If you forgot to set the *ProcessorType* column to be NULL (NULLABLE = YES), we can run an SQL ALTER TABLE command to fix that, as shown below.

15. We need to check the data in the new *computer* table. Right-click the **computer table object**, and then click the **Select Rows – Limit 1000** command. The data in the computer table is displayed, as shown in Figure C-34. All the data is correct.

16. If you need to modify the computer table to match the COMPUTER column characteristics in Step 6, here is how to set the NULL/NOT NULL setting of ProcessorType. We also need to add the two CHECK CONSTRAINTS.

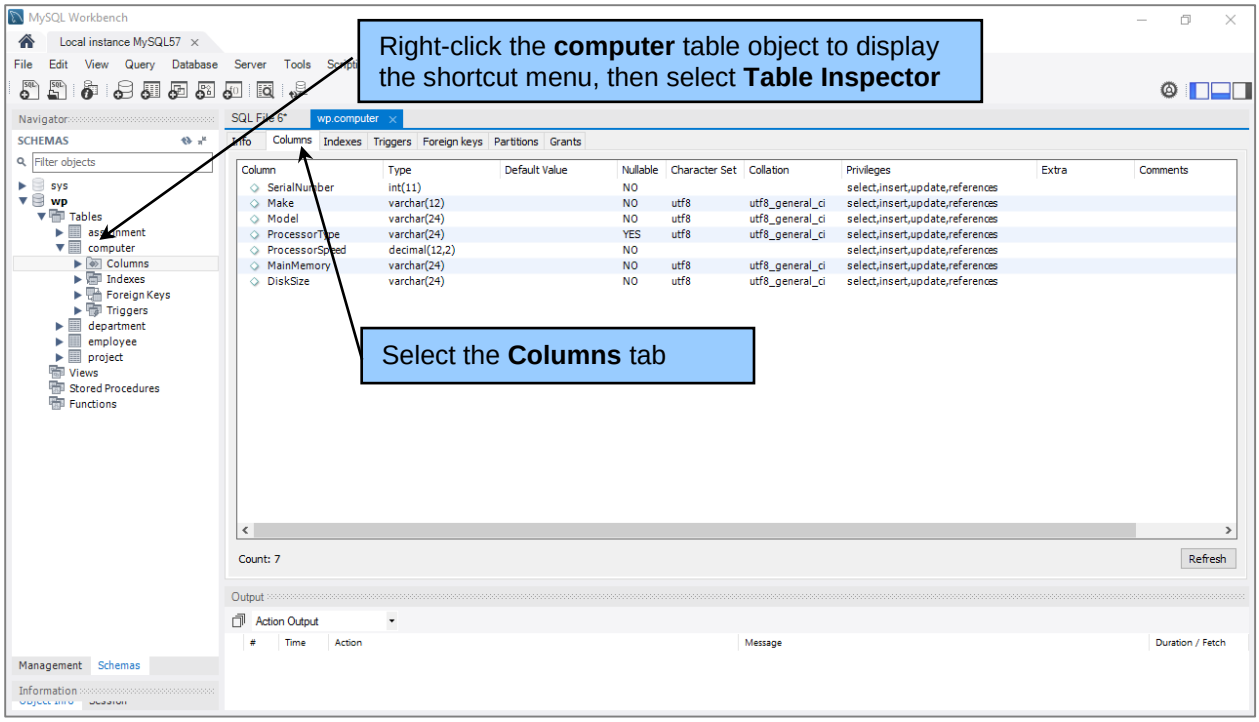


Figure C-33 — The COMPUTER Table Characteristics

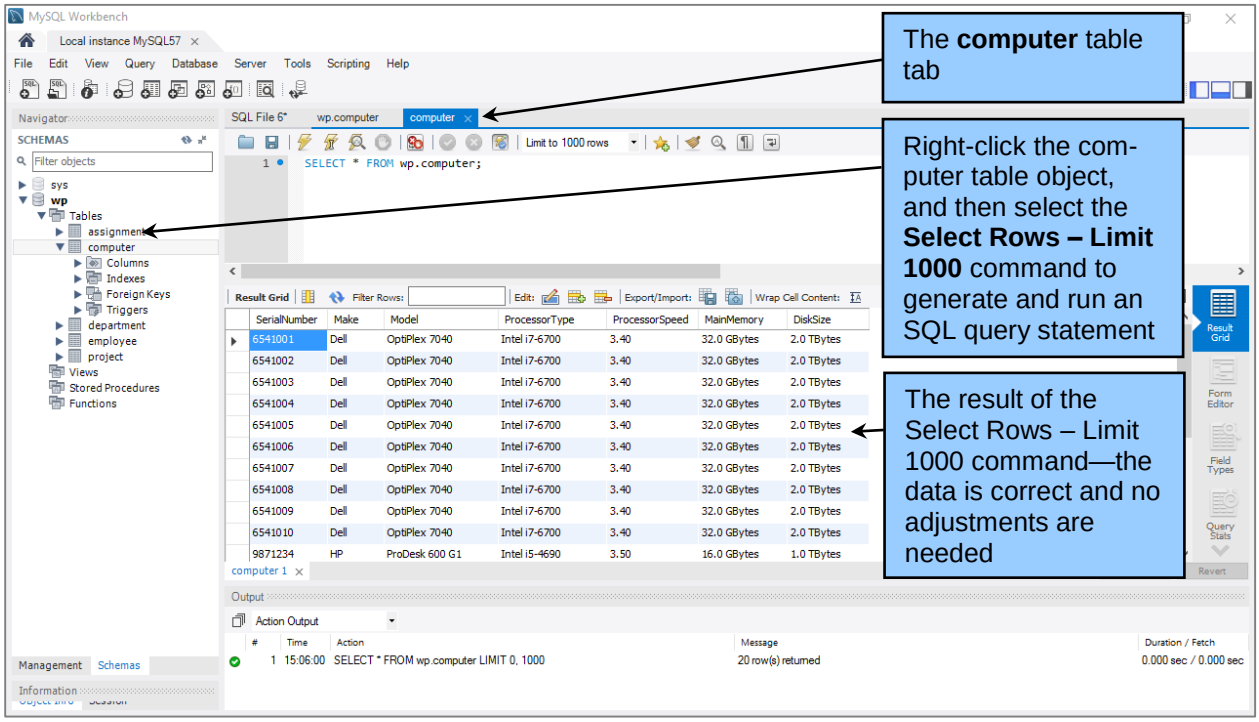


Figure C-34 — The Final COMPUTER Table Data

17. To set the NULL/NOT NULL status of ProcessorType requires the use of a MYSQL particular syntax. If needed, in the WP SQL query window write the SQL-ALTER-TABLE-AppC-01 statement:

```
/* *** SQL-ALTER-TABLE-AppC-01 *** */
ALTER TABLE COMPUTER
      CHANGE COLUMN ProcessorType ProcessorType VARCHAR(24) NULL;
```

18. Some DBMS products allow us to ensure that columns contain certain data values. For example, to ensure that the Make column contains only 'Dell', 'Gateway', 'HP', or 'Other'. However, MySQL allows the command to run, but does not enforce these "CHECK" values. To enforce the integrity of our data MySQL requires us to use triggers, which are beyond the scope of this appendix. For the curious, here is the syntax to set the CHECK CONSTRAINT for the computer make, in SQL-ALTER-TABLE-AppC-02 statement:

```
/* *** SQL-ALTER-TABLE-AppC-02 *** */
ALTER  TABLE COMPUTER
      Add CONSTRAINT MAKE_CHECK CHECK
            (Make IN ('Dell', 'Gateway', 'HP', 'Other'));
```

19. The COMPUTER table has now been added to the WP database. Close the 'computer' table tab.

How Do I Create User Accounts in a MySQL 5.7 Database?

When you, as a database administrator, create a database in MySQL 5.7, you have full permissions within the DBMS to do whatever you need to with that database. However, in order for other people (or applications) to use that database, you must create appropriate user accounts with logins and appropriate permissions.

A full discussion of database administration and database security is provided in Chapter 6. Here, we will discuss the actual steps taken in a MySQL 5.7 database to create a user account and give it the appropriate permissions to read data from and write data to the WP database. We will then use the new user account as we demonstrate how to link a Microsoft Access 2016 database to a MySQL Server 5.7 database using the ODBC Connector.

Creating a WP User Login with Permissions:

1. In the MySQL Workbench, click the **Server | Users and Privileges** commands. The Users and Privileges screen is shown in Figure C-35.
2. Select the **Add Account** button, and enter the new userid **WP-User** and the password **WP-User+password**, as shown in Figure C-36. Click the **Apply** button.
3. Select the Schema Privileges tab at the top, then click the **Add Entry**… button as shown in Figure C-37. The **New Schema Privilege Definition** dialog box is displayed. Select the **Selected Schema** radio button, then select **wp** in the drop-down list at the right. Click the **OK** button.

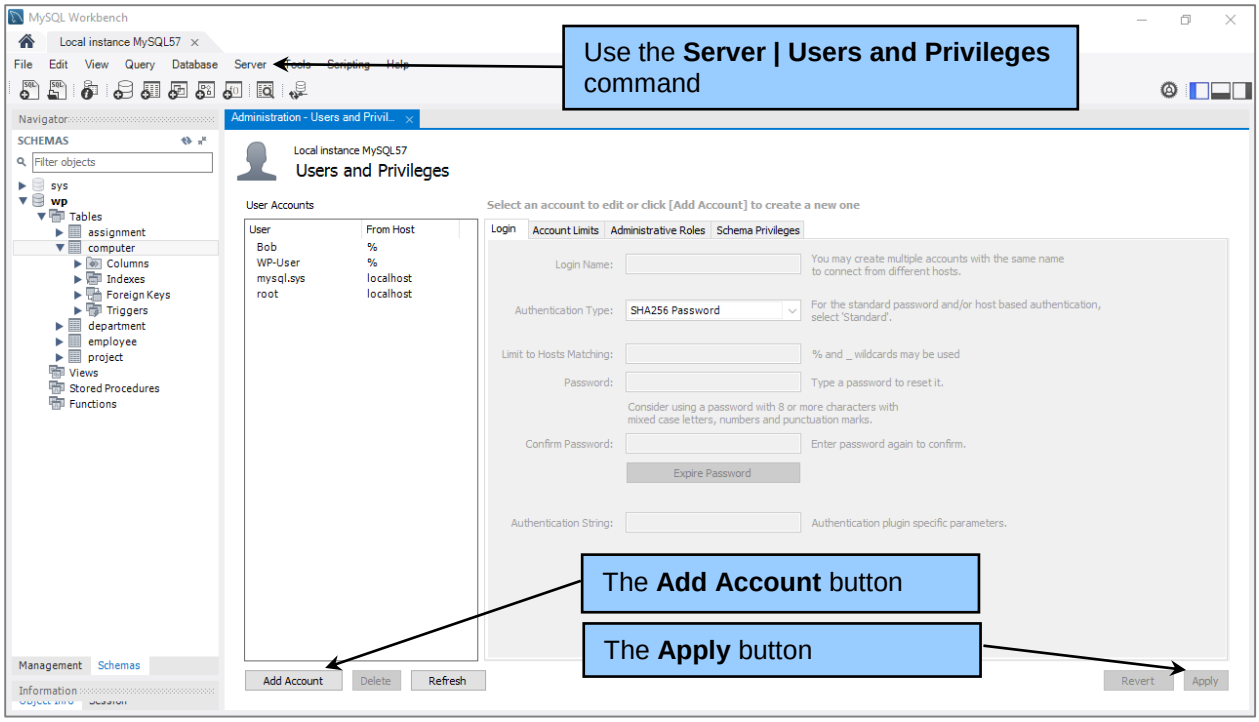


Figure C-35 — The MySQL 5.7 Workbench Users and Privileges screen

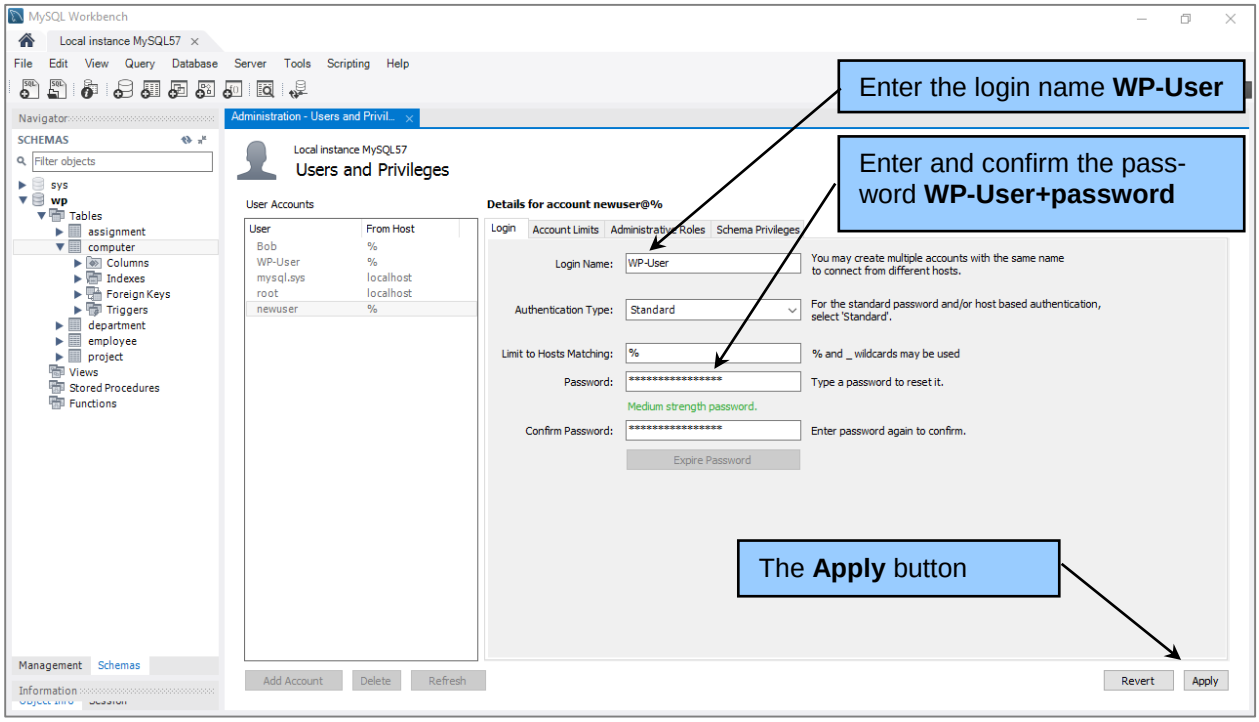


Figure C-36 — The MySQL 5.7 Workbench Add User screen

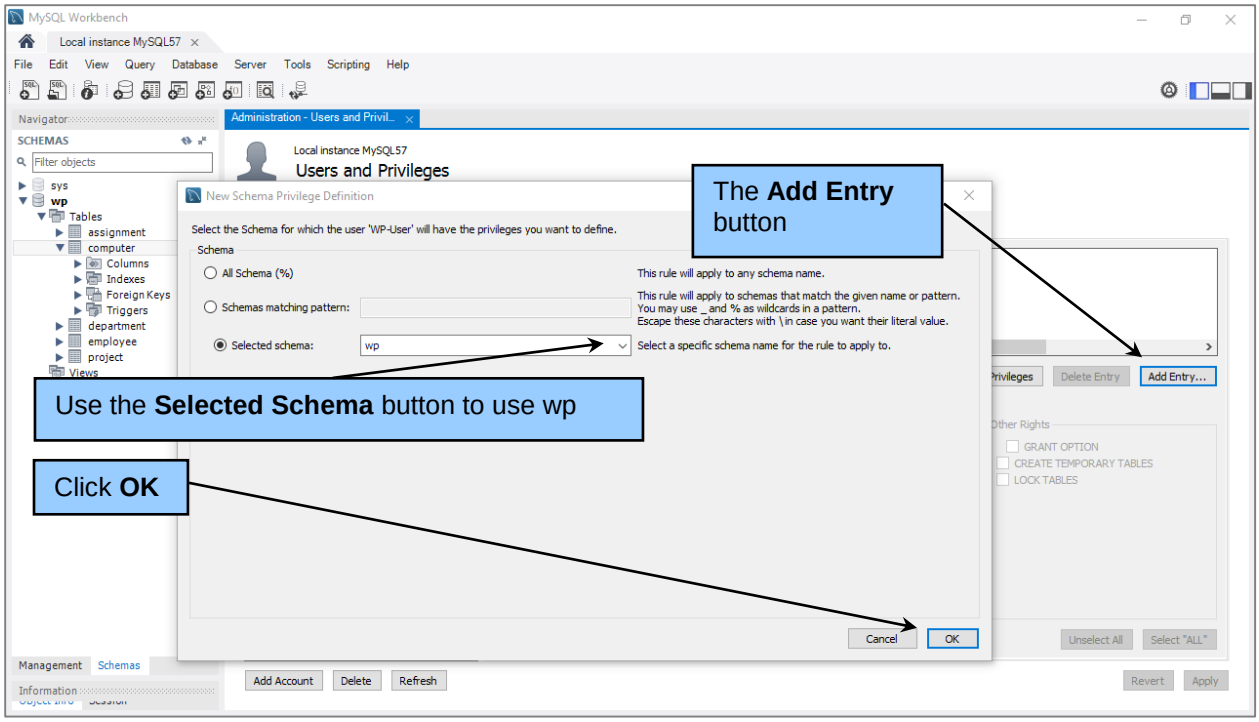


Figure C-37 — The MySQL 5.7 Workbench Add New Schema Privilege screen

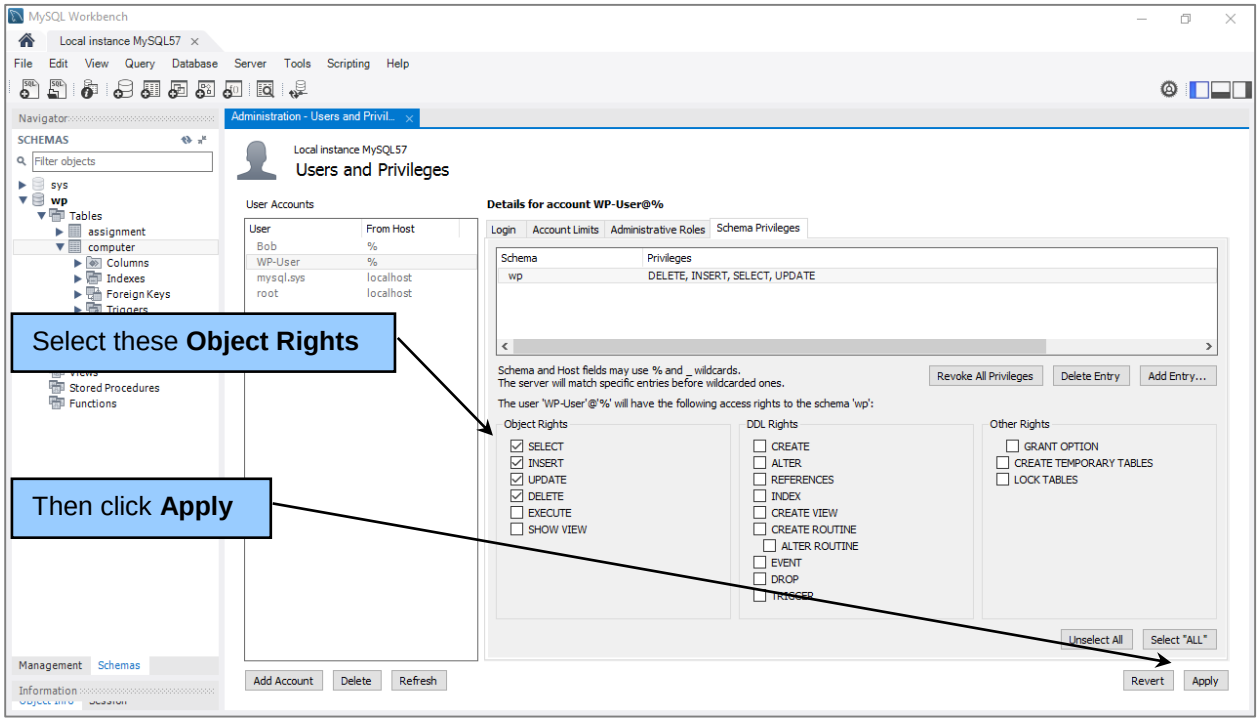


Figure C-38 — The MySQL 5.7 Workbench Select Privileges Screen

4. A list of Object Rights, DDL Rights, and Other Rights is displayed. In practice, you should select only those operations that you want the wp-user to perform. It is the best policy to allow only those operations required by that user to perform their role. In this case we will select only the Object Rights of SELECT, INSERT, UPDATE, and DELETE to allow wp-user to perform these actions on data in the databae, but we will not give wp-user the permissions needed to not modify the WP database structure. Select those options now, as shown in Figure C-38, and then click the **Apply** button.
5. The new WP-User account is created. We will use it for the following ODBC section. Use the **File | Exit** command to close MySQL Workbench.

How Do I Create an ODBC Connection from Microsoft Access 2016 to a MySQL Server 5.7 Database?

While MySQL Server 5.7 is an excellent enterprise-class DBMS, it does not provide any application development tools. Microsoft Access 2016 does provide a set of application development tools such as forms, reports, stored queries, and menu systems (see Appendix H, "The Access Workbench—Section H—Microsoft Access 2016 Switchboards"). Thus, it would be useful to have a way to use Microsoft Access 2016 as the application development frontend for a MySQL Server 5.7 database. Also, since most companies store their data on a centralized server, being able to use Access to connect to remote data will be very helpful for your workplace productivity.

This is actually very easy to do using an **Open Database Connectivity (ODBC)** link. For a full discussion of ODBC and how to use it in Web-based database applications, see Chapter 7. Here, we will simply walk through the steps necessary to build and use the connection.

We will continue to use the WP database that we have been using, and we will create a Microsoft Access 2016 database to act as the application development environment for the WP database. We will name our Microsoft Access 2016 database as WPIS.accdb (for Wedgewood Pacific Information System).

Creating the WPIS.accdb Database:

1. Open **Microsoft Access 2016**.
2. Click the **Blank desktop database** icon to open the Blank desktop database dialog box.
3. In the **Blank database** dialog box, use the folder icon to navigate to the desired directory, then type in the file name **WPIS.accdb**, click **OK**, then then click the **Create** button.
4. The new WPIS.accdb database is displayed.
5. Close the open **Table1** tabbed window.

Because we will be connecting to a DBMS that uses **ANSI standard SQL (ANSI 92)** instead of **Microsoft ANSI-89 SQL** (this topic is discussed in depth at the beginning of Chapter 3), we need to set the Microsoft Access 2016 options for WPIS.accdb to working with ANSI standard SQL.

Setting the Microsoft Access 2016 SQL Setting:

1. Click the **File** command tab, and then click the **Options** button. The Access Options dialog box is displayed.
2. In the **Access Options** dialog box, click the **Object Designers** button to display the Object Designers page.
3. In the **Object Designers** page, find the **SQL Server Compatible Syntax (ANSI 92)** section of the Query Design settings. Click the **This database checkbox** as shown in Figure C-39. MySQL uses the new standard as well as SQL Server.
4. Click the **OK** button to save the settings and close the dialog box.

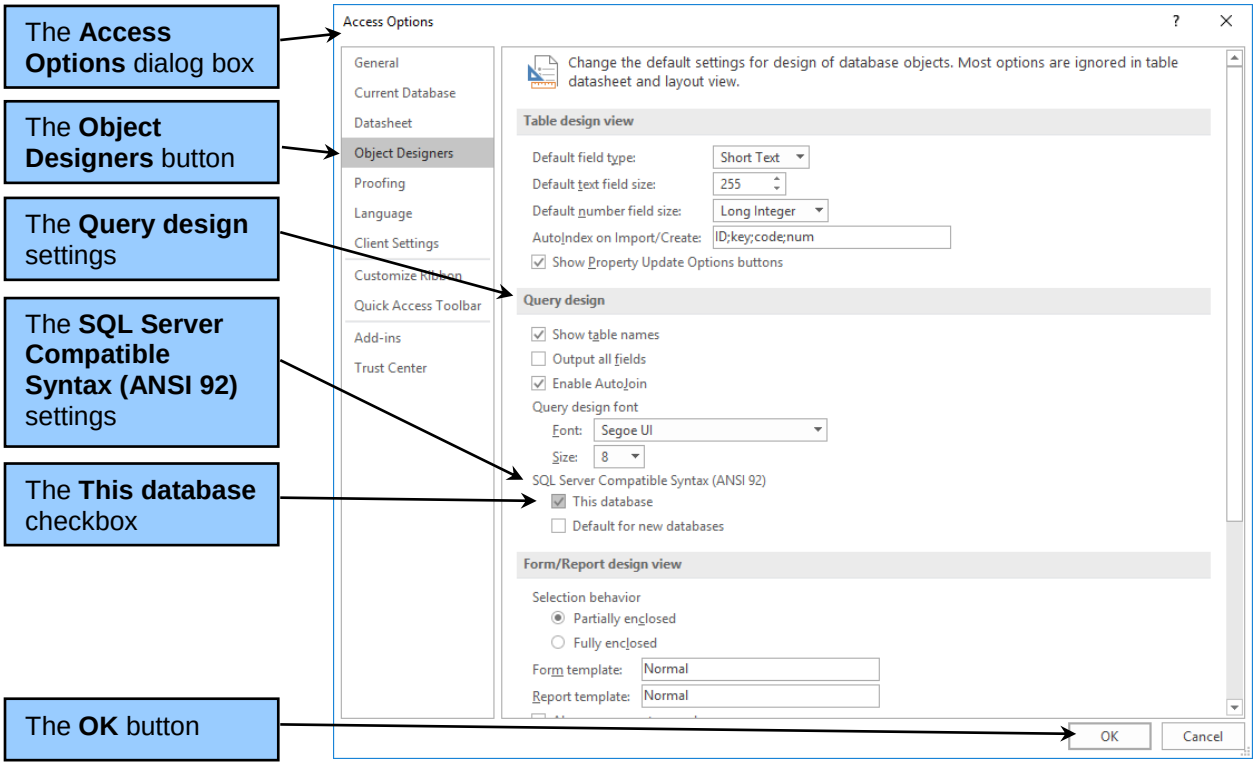


Figure C-39 — The SQL Server Compatible Syntax (ANSI 92) Checkbox

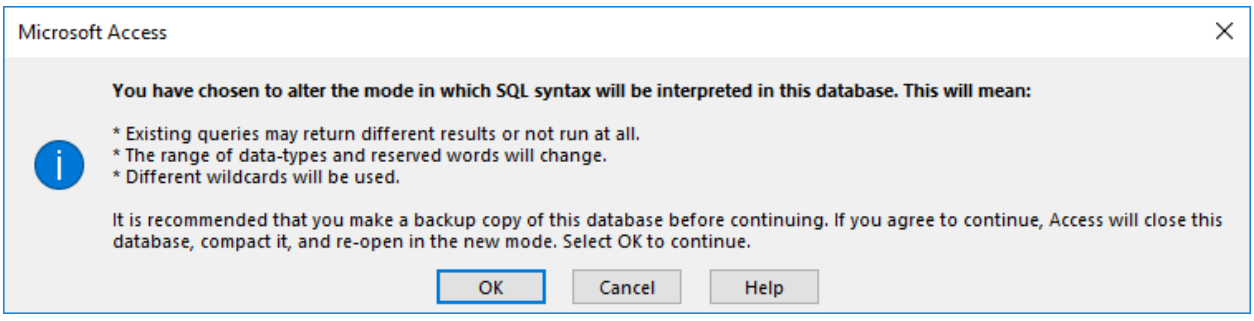


Figure C-40 — The Microsoft Access Information Dialog Box

5. A Microsoft Access Information dialog box is displayed, as shown in Figure C-40.
6. Read the information in the dialog box, and then click the **OK** button
7. Microsoft Access takes the actions discussed in the information dialog box and then reopens the WPIS.accdb database with the **Security Warning message bar** displayed.
8. In the Security Warning message bar, click the **Enable Content** button.

The WPIS database is now ready to use. Our next step is to connect to the WP database in the MySQL 5.7 server on your computer. To do this, we create a link to data external to Microsoft Access by using a ODBC data source. For a full discussion of ODBC data sources (each of which is called a **DSN**), see Chapter 7. And in creating the ODBC DSN, we will make use of the WP-User login we previously created.

Linking the Microsoft Access 2016 Database to an External Data Source via ODBC:

1. In the Microsoft Access 2016 WPIS.accdb database, click the **External Data** command tab, and then click the **ODBC Database** button in the Import & Link commands section.
2. The **Get External Data - ODBC Database Wizard** dialog box is displayed, as shown in Figure C-41.
3. In the Get External Data - ODBC Database Wizard dialog box, the *Select the source and destination of the data* page is displayed. Click the **Link to the data source by creating a linked table** radio button also shown in Figure C-41, and then click the **OK** button.
4. The **Select Data Source** dialog box is displayed, as shown in Figure C-42. This is the dialog box that we will use to create the needed OBDC DSN.
5. In the Select Data Source dialog box, make sure the **Machine Data** Source tab is selected, and then click the **New** button to display the Create New Data Source dialog box as shown in Figure C-43. Select the **User Data Source** radio button, then the **Next** button.
6. In the Select a Driver dialog box, scroll down through the list of drivers until you can see the driver named **MySQL ODBC 5.3 ANSI Driver** as shown in Figure C-44. Click this driver name to select it, and then click the **Next** button.
7. Once the driver is selected, a confirmation box with the driver name is shown in Figure C-45.
8. Once the driver is selected, click the **Next** button in the Create New Source dialog box. You will be prompted to enter the name of your data source (wp-source), the user (WP-User), the password (WP-User+password), and the database (wp). Click Next, as shown in Figure C-46.
9. The Select Data Source – Machine Data Source dialog box is displayed, as shown in Figure C-47. Select wp-source then click the **OK** button.
10. Finally, we get the Link Tables selection box, as shown in Figure C-48. These are the tables that we want linked to our local Access database. Click **Select All** to link to all the tables in WP, then click the **OK** button.
11. The linked tables should appear in the Objects list pane on the left. You can then run queries using them, or create forms for data entry. Try running a simple query now, like “SELECT * FROM Employee”. The query will run against table data on MySQL server and return the result to you.

Congratulations, you have completed the ODBC section! The last section will guide you in using the Workbench to create database designs.

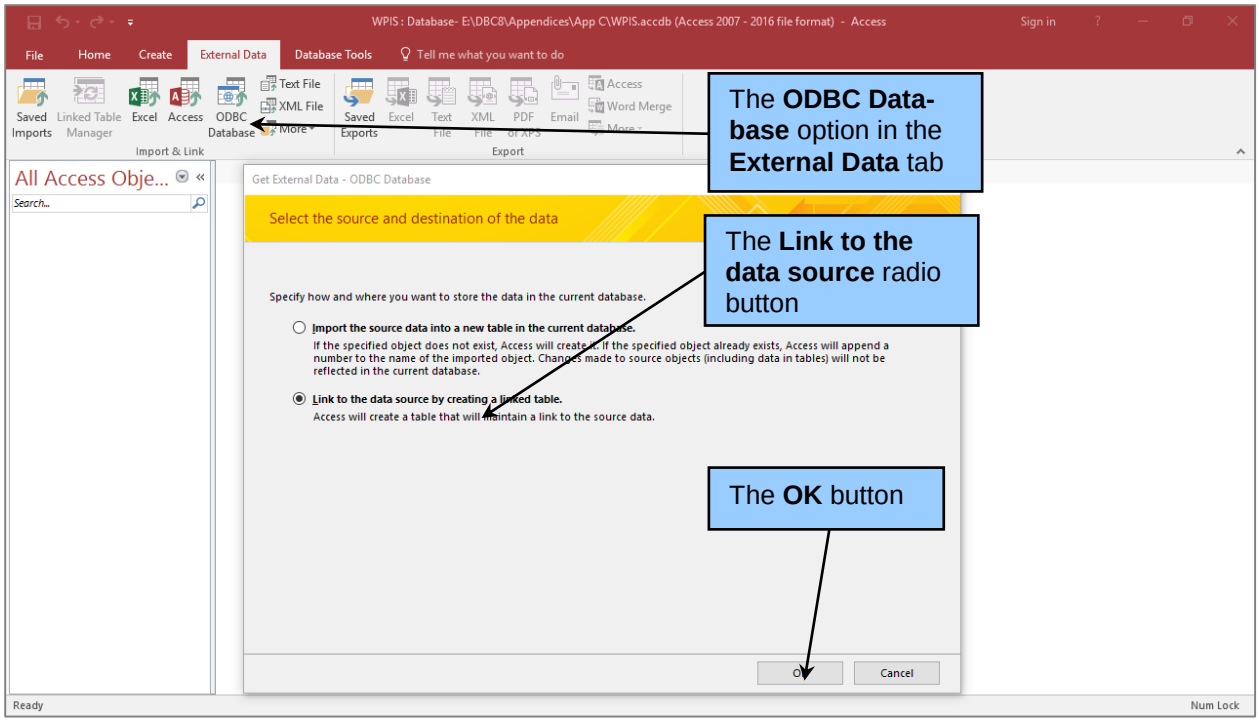


Figure C-41 — The Link to the data source Dialog Box

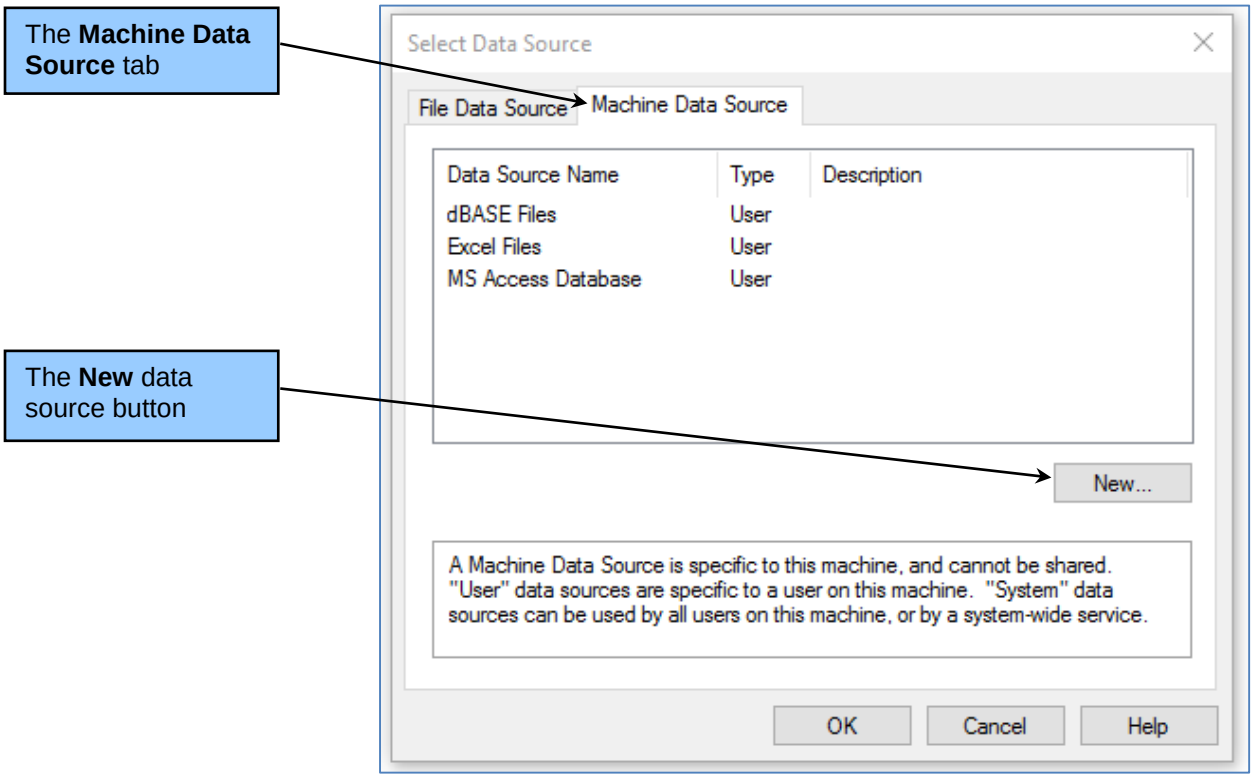


Figure C-42 — The Select Data Source Dialog Box

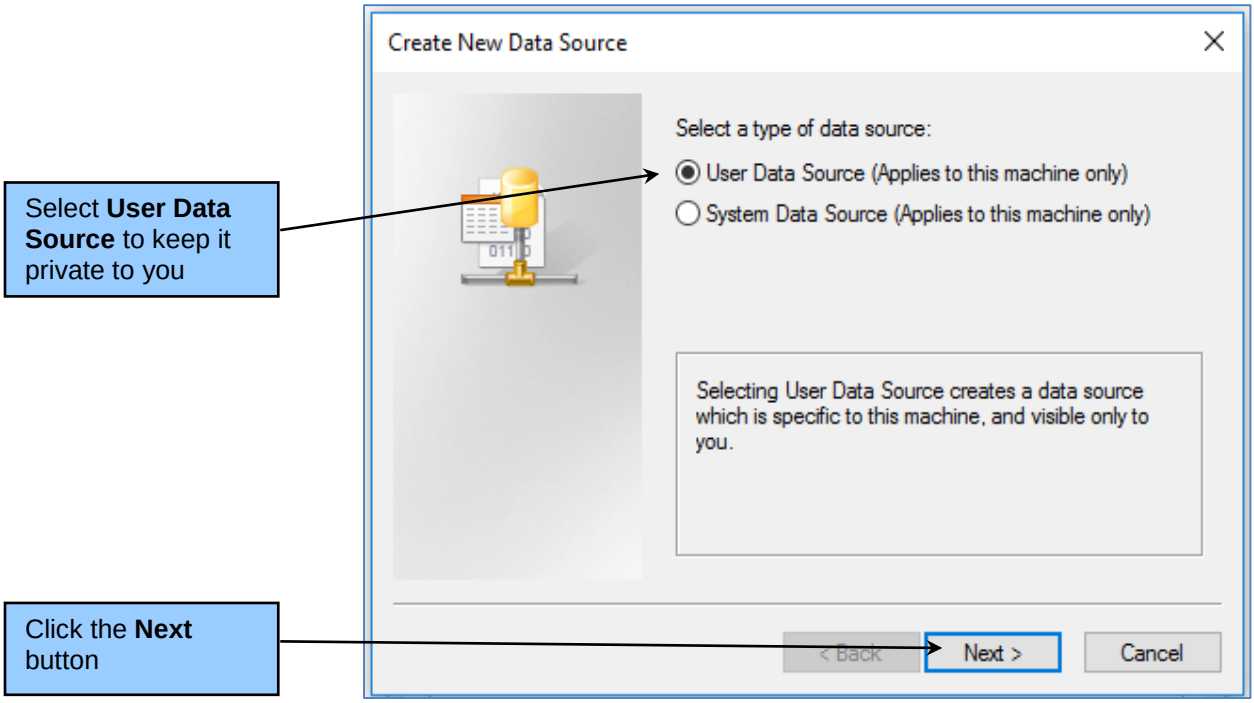


Figure C-43 — The Select a Type of Data Source Dialog Box

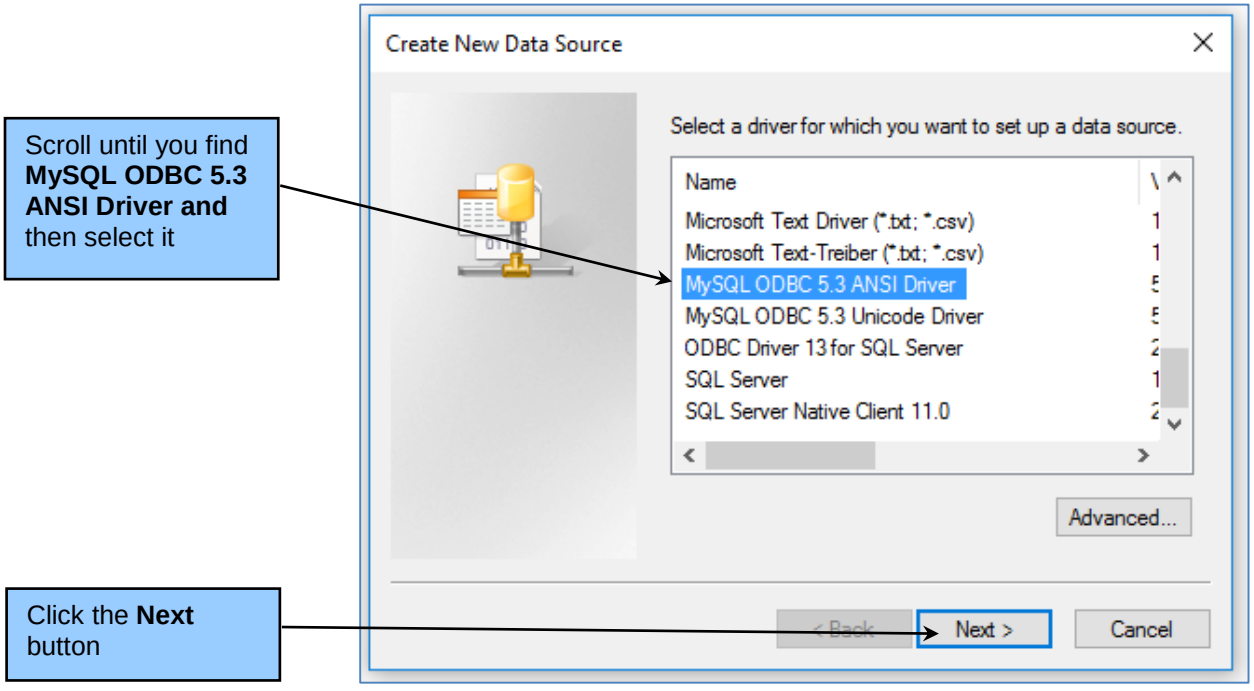


Figure C-44 — The Select a Driver Dialog Box

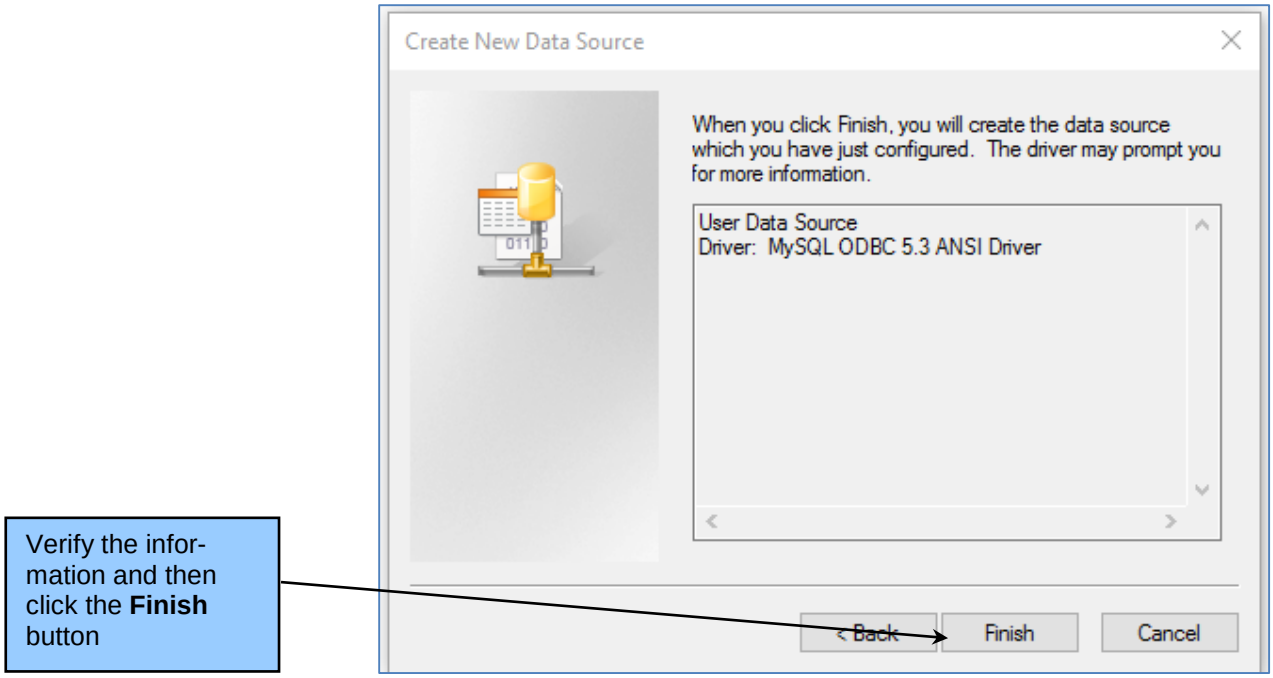


Figure C-45 — Create New Data Source Dialog Box – Confirmation

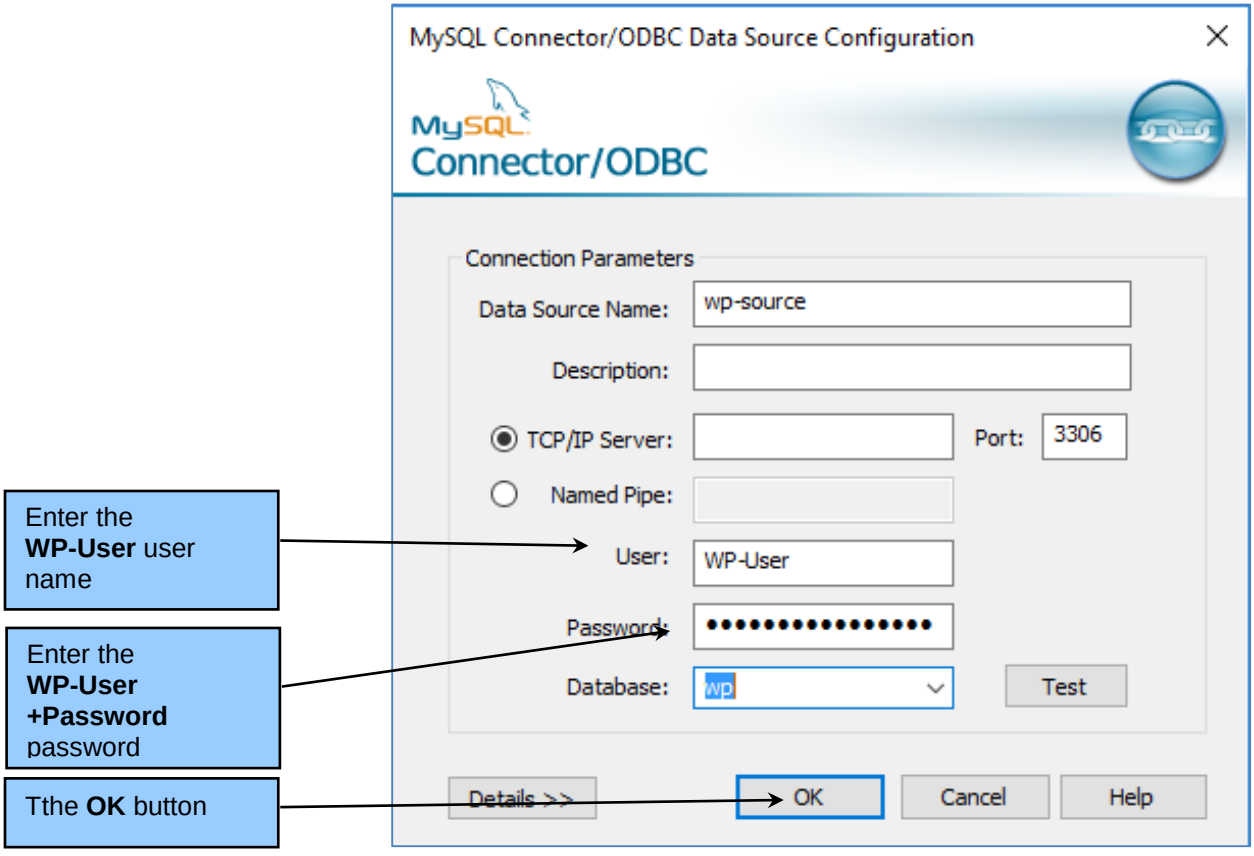


Figure C-46 — ODBC Connection Parameters

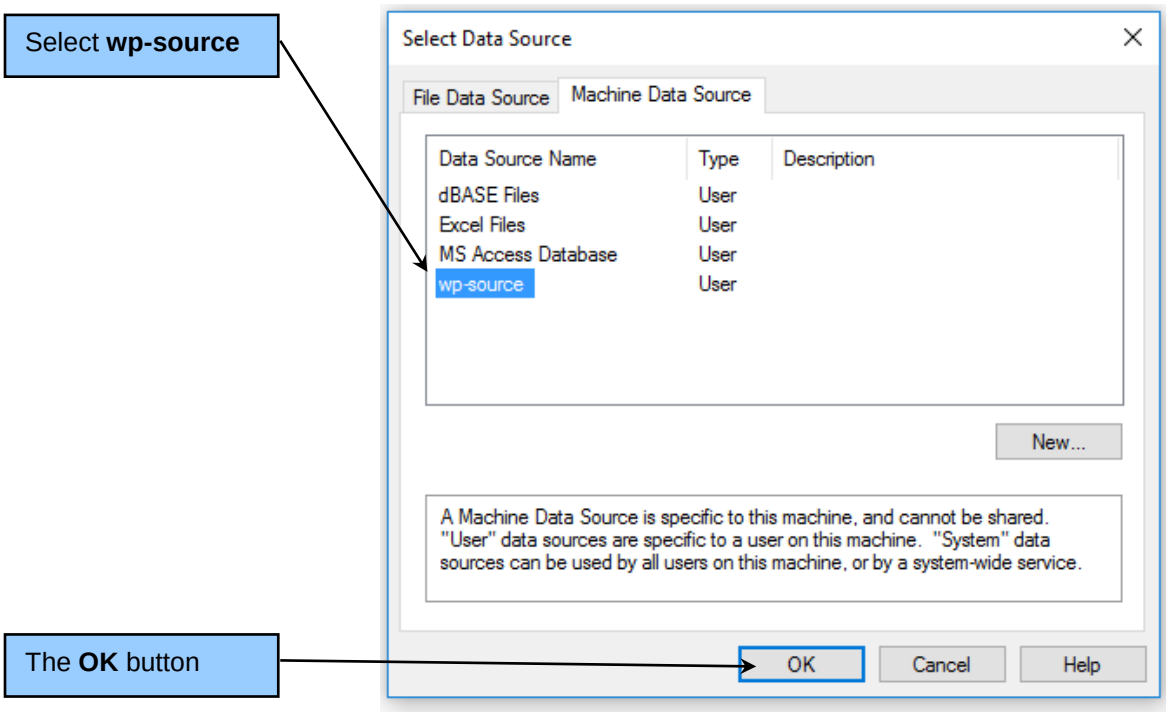


Figure C-47 — ODBC Machine Data Source Configuration with wp-Ssource Added

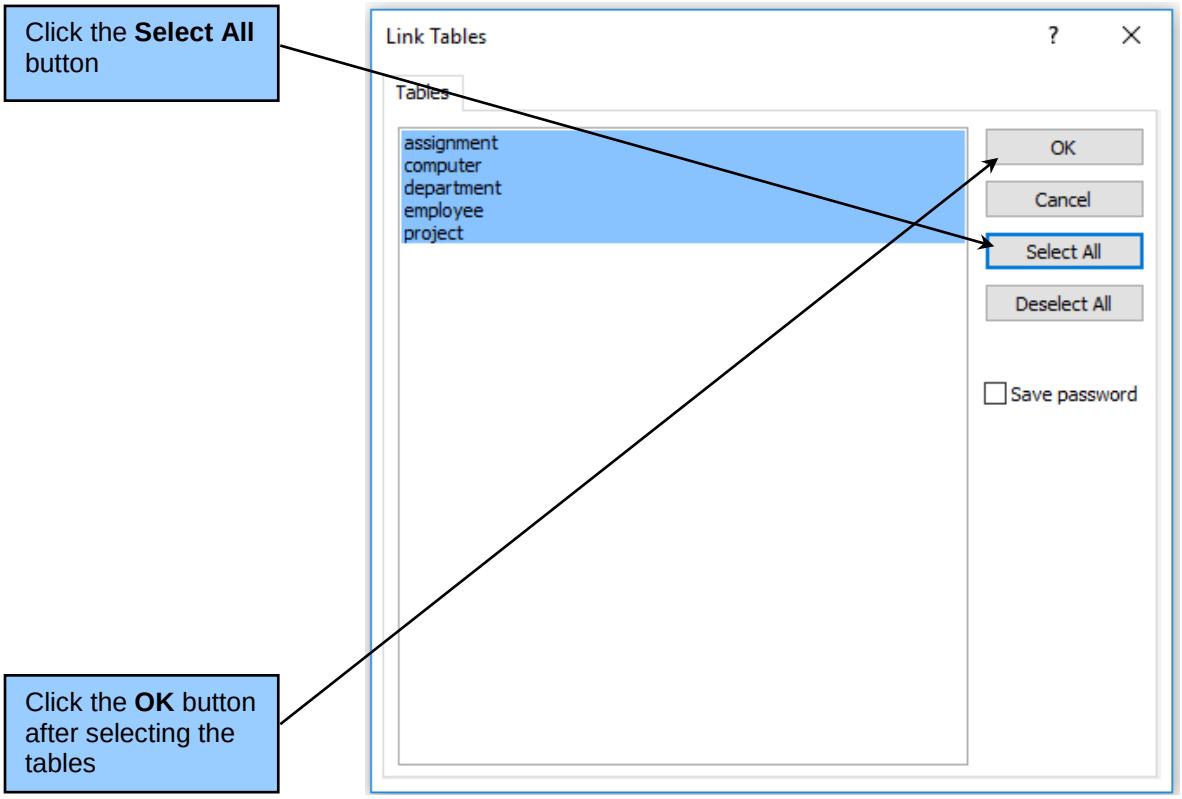


Figure C-48 — Access Link Tables Selection Dialog Box

How Do I Create Database Designs in the MySQL Workbench?

As we discuss in Chapters 4 and 5, a **data model** is a logical or conceptual view of the database. A **database design** defines the database characteristics that will be implemented in the actual database. One of the main differences between a data model and a database design is how N:M relationships are handled. In a data model, N:M relationships exist as N:M non-identifying relationships between two strong entities. In a database design, N:M relationships are broken into two 1:N identifying relationships between three ID-dependent entities.

While the MySQL Workbench refers to "data modeling" capabilities, these are really database design capabilities. The MySQL Workbench cannot create true N:M relationships, only the two 1:N relationships between three entities (the two original entities in the true data model and an intersection table).

We will illustrate creating database designs in the MySQL Workbench by creating an E-R diagram for the Wedgewood Pacific (WP) database that we created earlier in this appendix.[8] When we are done, we will have a database diagram that will closely resemble Figures 5-16 and 5-17.

How Do I Create a Database Model and E-R Diagram in the MySQL Workbench?

To create a new data model in the MySQL Workbench, you can:

- Use the **File | New Model** command, or
- Click the **New Model** button on the initial Home page (looks like Models (+)).

Once the model is created, entity-relationship (E-R) diagrams, which the MySQL Workbench refers to as *EER* for the **extended entity-relationship** model (which, as discussed in Chapter 4, is the correct term, although today the term *E-R model* always means the EER model), are created within the model.

Note that you can create database designs *without* connecting to a MySQL server. In our case, we have already connected to the MySQL server because we created the WP database.

Creating a New MySQL E-R (EER) Diagram:

1. Create a new folder in Documents called MySQL Workbench, then create a subfolder in it called EER-Diagrams.
2. Open the MySQL Workbench program from scratch.

[8] Of course, it could be argued that we really should have created the database design first and then implemented that design. In many database courses, the data modeling and database design topics (which we cover in Chapters 4 and 5) are taught before using SQL to create the databases (which we cover in Chapter 3). In this case, the database design will follow the actual implementation of the database in the DBMS. We prefer to introduce SQL earlier. There are two reasons for this. First, users who are never involved in creating databases still often use SQL or QBE for querying databases (usually in data warehouses or data marts as discussed in Chapter 8) to gather information. Second, we like to get our students involved with DBMSs, databases, and SQL as early in the course as possible. Either approach works, and your professor will choose the one that he or she prefers.

3. Click the **File | New Model** command. When the MySQL Model tabbed window is displayed, as shown on Figure C-49, close the Modeling Additions pane by using the button indicated in the figure.
4. In the MySQL Model tabbed window, click the **File | Save Model As…** command to display the **Save Model** dialog box. Browse to the Documents\MySQL Workbench\EER-Diagrams folder, and click the **New folder** button. Name the new folder **WP-Database**.
5. In the WP-Database folder, save the new model as **WP-Database-Design.mwb** (note that **.mwb* is the default file extension used by the MySQL Workbench for data models).
6. Double click the **Add Diagram** icon in the Model Overview section of the MySQL Model page, as shown in Figure C-49. A new, blank EER Diagram page is displayed, as shown in Figure C-50. Again note that MySQL uses the acronym *EER* where we are using *E-R*.
7. Close the Modeling Additions pane, and then click the **Save Model to Current File** button (see Figure C-50) to save the WP-Database-Design model with the E-R diagram.

Now that we have a blank E-R diagram work area available, we can build the E-R diagram itself. We start by adding a table to the E-R diagram. We will add the DEPARTMENT table. By looking at the SQL statement for this table below, we can see the columns that are used in the DEPARTMENT table.

Here is the SQL statement that creates the DEPARTMENT table (copied from Figure C-16):

```
CREATE  TABLE DEPARTMENT(
       DepartmentName      Char(35)            NOT NULL,
       BudgetCode          Char(30)            NOT NULL,
       OfficeNumber        Char(15)            NOT NULL,
       DepartmentPhone     Char(12)            NOT NULL,
       CONSTRAINT          DEPARTMENT_PK       PRIMARY KEY(DepartmentName)
       );
```

Creating a Table in the MySQL E-R (EER) Diagram:

1. In the E-R diagram toolbar, click the **Place a New Table** button as shown in Figure C-50.
2. Move the cursor over the blank E-R diagram area. As you do so, notice that table property options are displayed across the top of the design area, as shown in Figure C-51. We do not need to change any of these.
3. When the cursor is at the place where you want the table icon, click the left mouse button. A new table object named table1 is created on the E-R Diagram, as shown in Figure C-52.
4. Double-click the **table1** object to open the MySQL table editor for table1 as shown in Figure C-53. Note that the table editor has tabs along the bottom of the editor window and that the Columns tab is selected so that the Columns pane is displayed.
   - **NOTE:** Alternatively, right-click the table object to display a shortcut menu, and then click **Edit {TableName} …** .
5. In the Columns pane, click the **Table Name** text box to select it if needed, and then type in the table name **DEPARTMENT**. Note that the new table name immediately appears on the table editor tab and the table object in the E-R diagram area, as shown in Figure C-54.

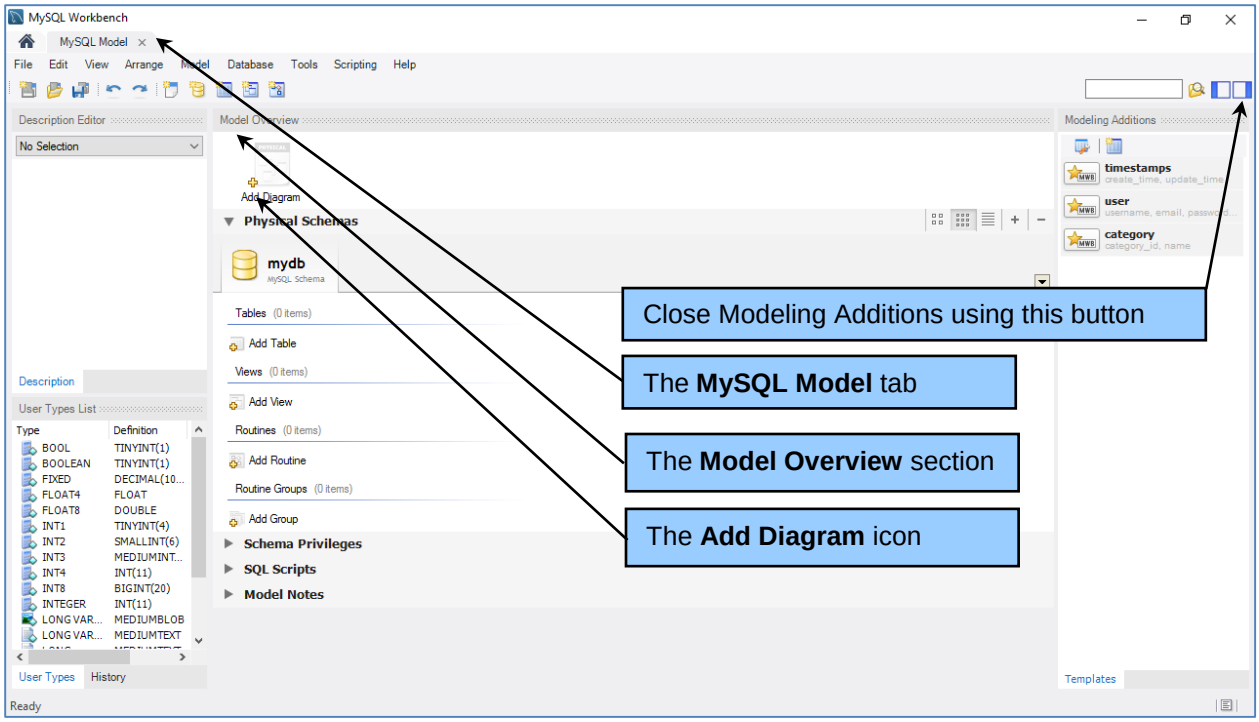


Figure C-49 — The MySQL Model Tab and Window

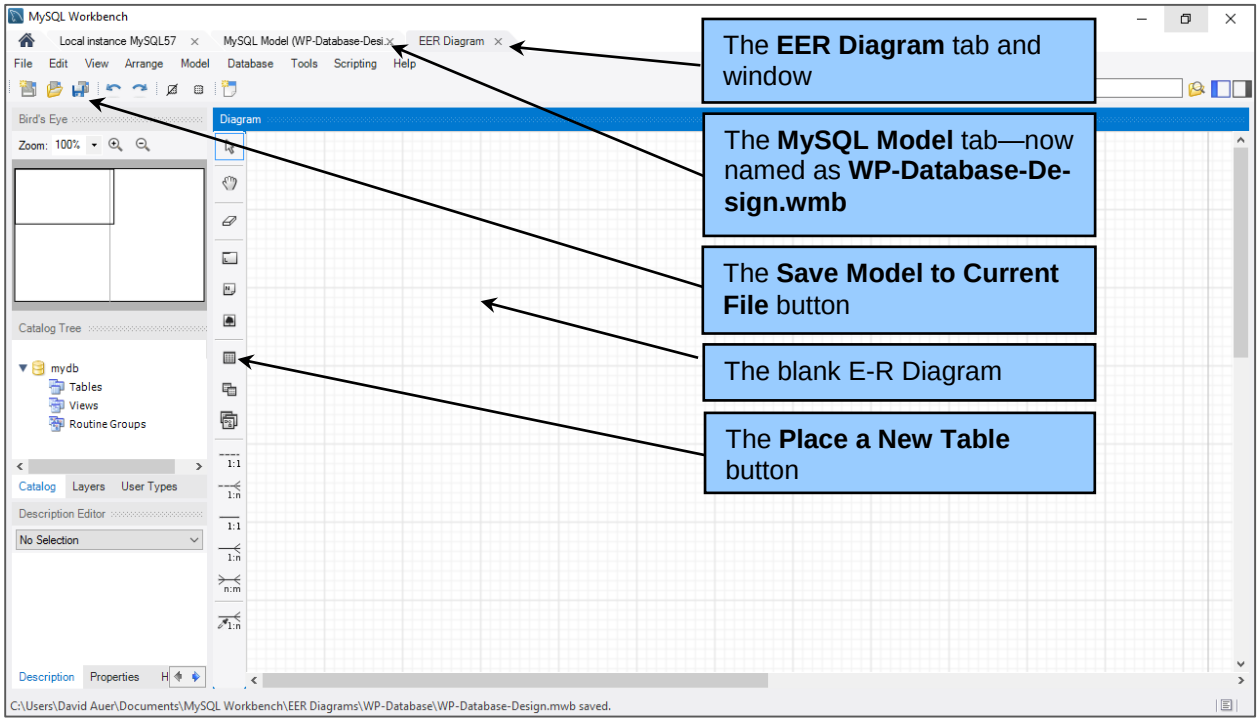


Figure C-50 — The Blank E-R Diagram

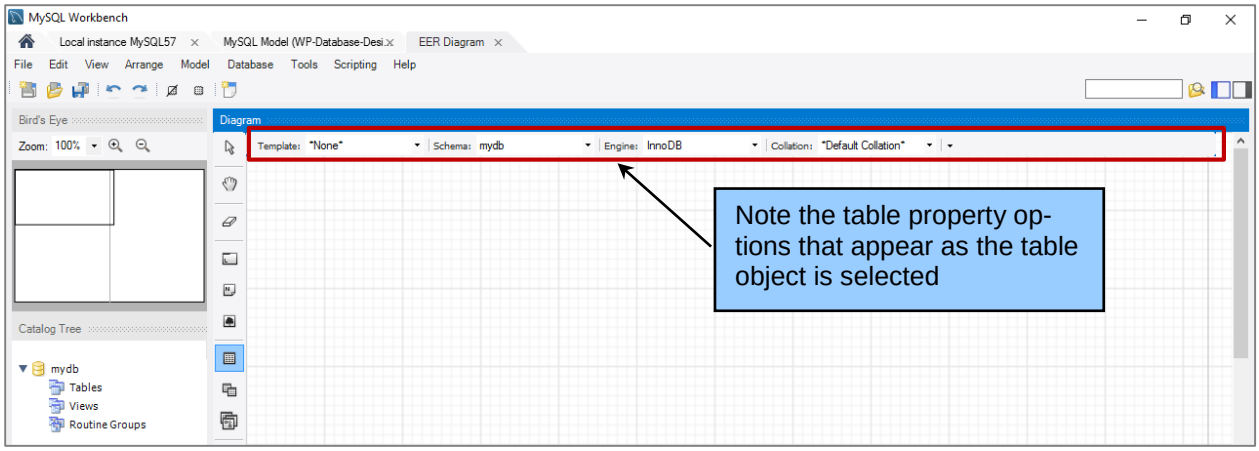


Figure C-51 — Placing the Table Object

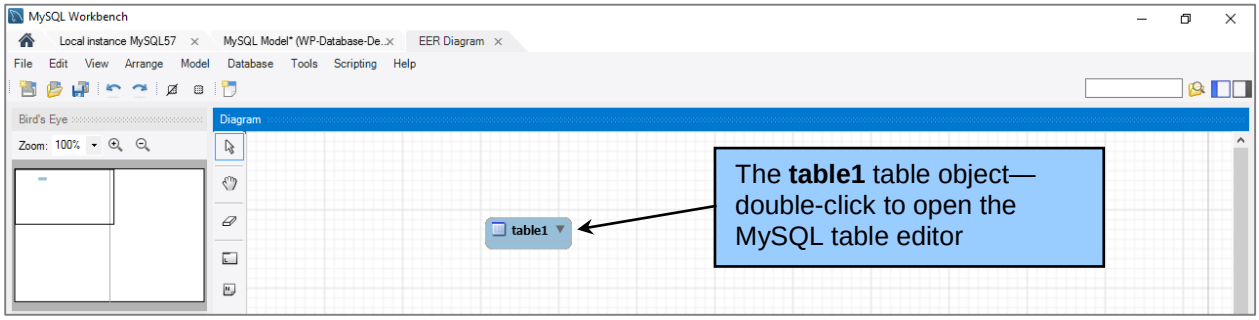


Figure C-52 — The table1 Table Object

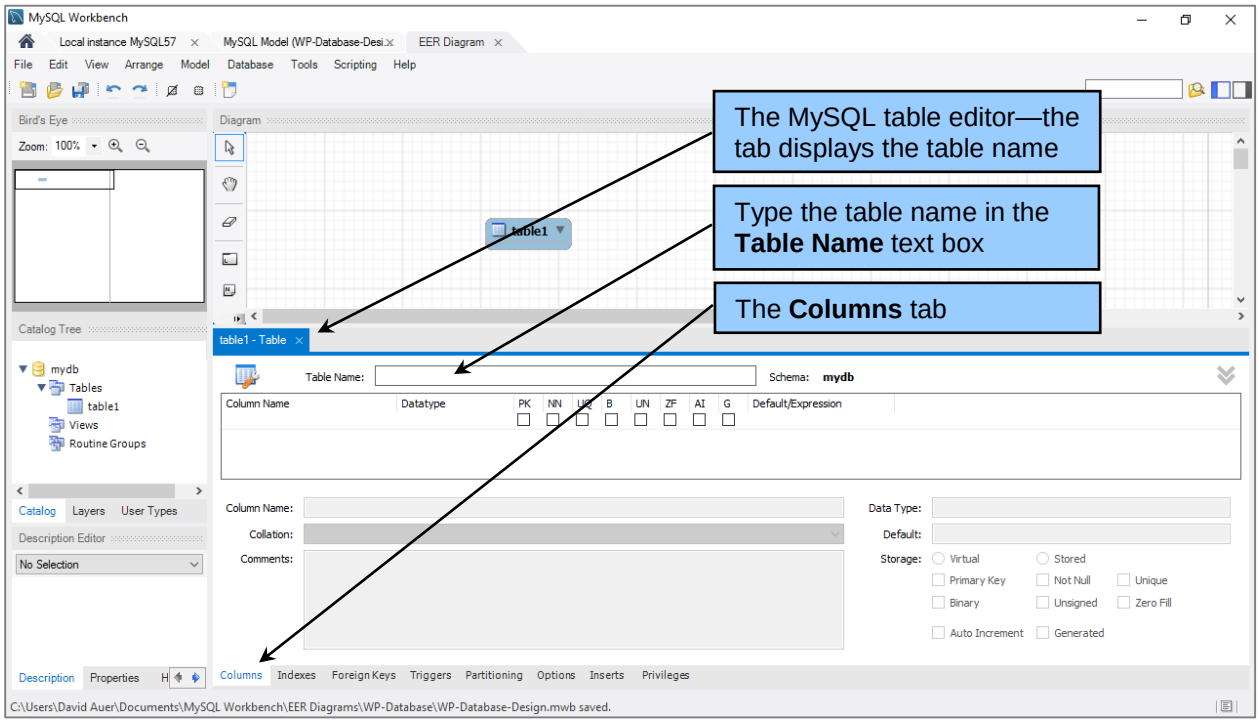


Figure C-53 — The MySQL Table Editor

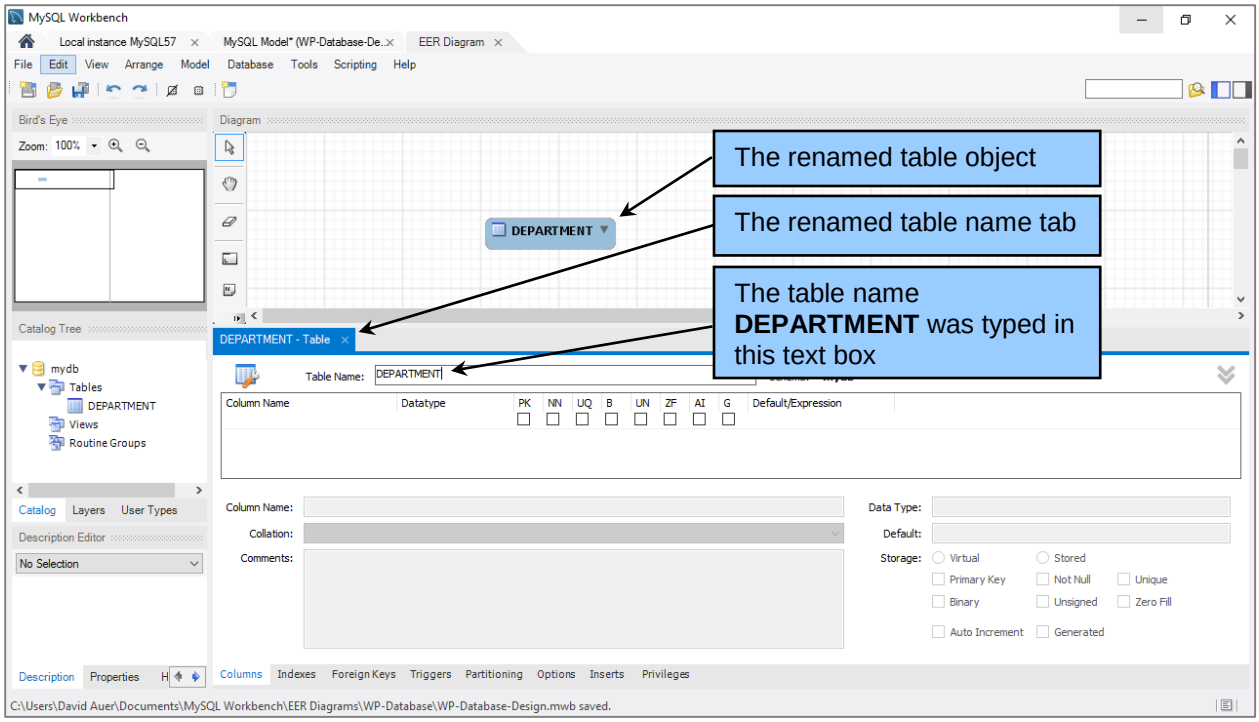


Figure C-54 — The Renamed Table

Now that we have the DEPARTMENT table created and named, we will create the columns in the table.

As shown in Figure C-55, MySQL uses checkboxes with the following abbreviations and a Default text box for defining column characteristics in the MySQL Table Editor:

- **PK** Primary Key – Check if this column is the primary key or part of a composite primary key.
- **NN** NOT NULL – Check if this column must have an inserted value.
- **UQ** UNIQUE – Check if this column must contain a unique value.
- **BIN** Binary – Check if this the column uses only two values, such as 0 and 1 or Yes and No.
- **UN** Unsigned Data Type – Check if this column uses numbers without negative values and you specifically want to not permit negative numbers where they might not otherwise be allowed. Zero fill (ZF) numbers are automatically checked UN.
- **ZF** Zero Fill – Check if this column should be automatically filled with zeros.
- **AI** AUTO_INCREMENT – Check if this column is a primary key that should have sequential surrogate key values.

The default text is used, of course, for specifying the DEFAULT value for a column if one is required. Figure C-58 shows the complete DEPARTMENT table. The key symbol indicates the primary key, and the filled-in, light blue diamonds indicate NOT NULL columns (NULL columns have an empty diamond).

Creating the DEPARTMENT Table Columns in the MySQL E-R (EER) Diagram:

1. In the MySQL Table Editor, double-click the row area immediately in the **Column Name** column in the Columns area. MySQL Workbench generates a primary key name idDEPARTMENT. Click the PK and NN columns to set the primary key. This primary key is also displayed in the DEPARTMENT table object, as shown in Figure C-55.
2. The correct primary key column name for DEPARTMENT (as shown in the CREATE TABLE DEPARTMENT SQL statement shown previously in this appendix) is DepartmentName. Edit the Column Name to read **DepartmentName**.
3. The Datatype for DepartmentName is currently INT, but it should be Char(35). Click the Datatype drop-down list arrow in the Datatype field to display the Datatype drop-down list as shown in Figure C-56. Select the Char( ) datatype from the list (it is at the bottom of the list and not visible in Figure C-56) and edit the datatype to read **Char(35)**.

   - **NOTE:** Alternatively, you can just type the correct data type into the Datatype field.

4. There are no other settings that need to be changed for the DepartmentName column. It is already marked as PK (Primary Key) and NN (NOT NULL).
5. Double-click the Column Name field in the blank row immediately below DepartmentName. The completed DepartmentName column row and a new blank column row are displayed as shown in Figure C-57. Note that the correct column settings are now displayed in the DEPARTMENT table object in the E-R diagram. Continue adding new columns and set the datatypes and attributes to complete the DEPARTMENT table as shown in Figure C-58.
6. Click the **Save Model to Current File** button to save the changes, and then close the MySQL Table Editor by clicking the X in the DEPARTMENT – Table tab at the top of the table editor pane.

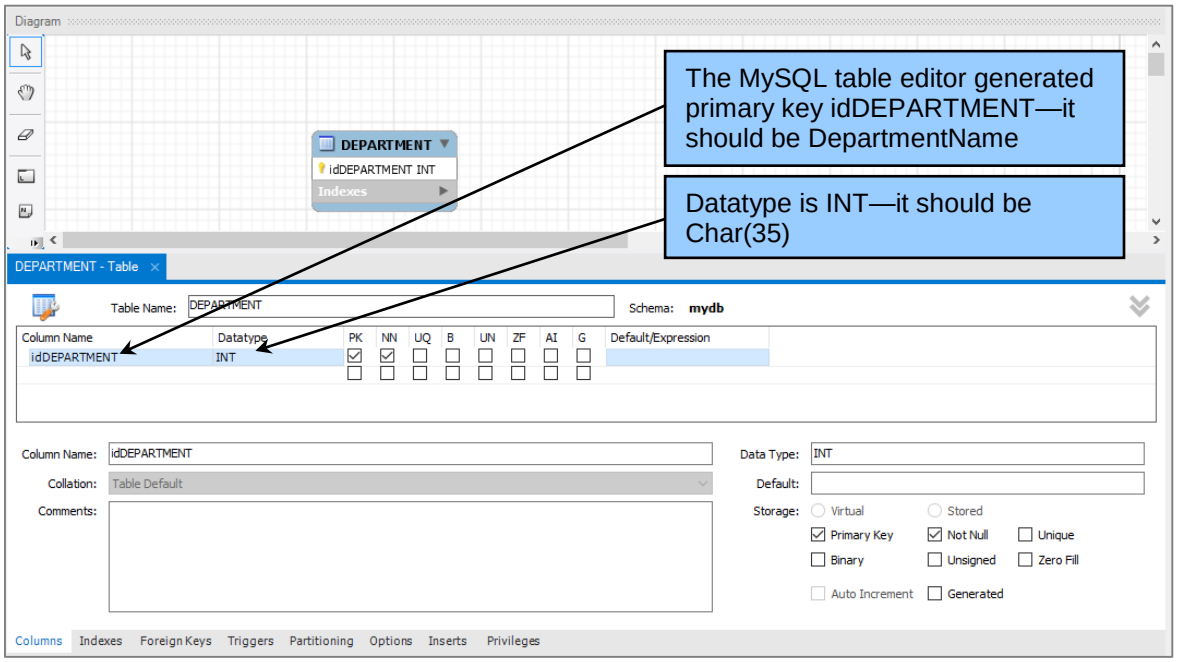


Figure C-55 — The MySQL Table Editor Generated Primary Key idDEPARTMENT

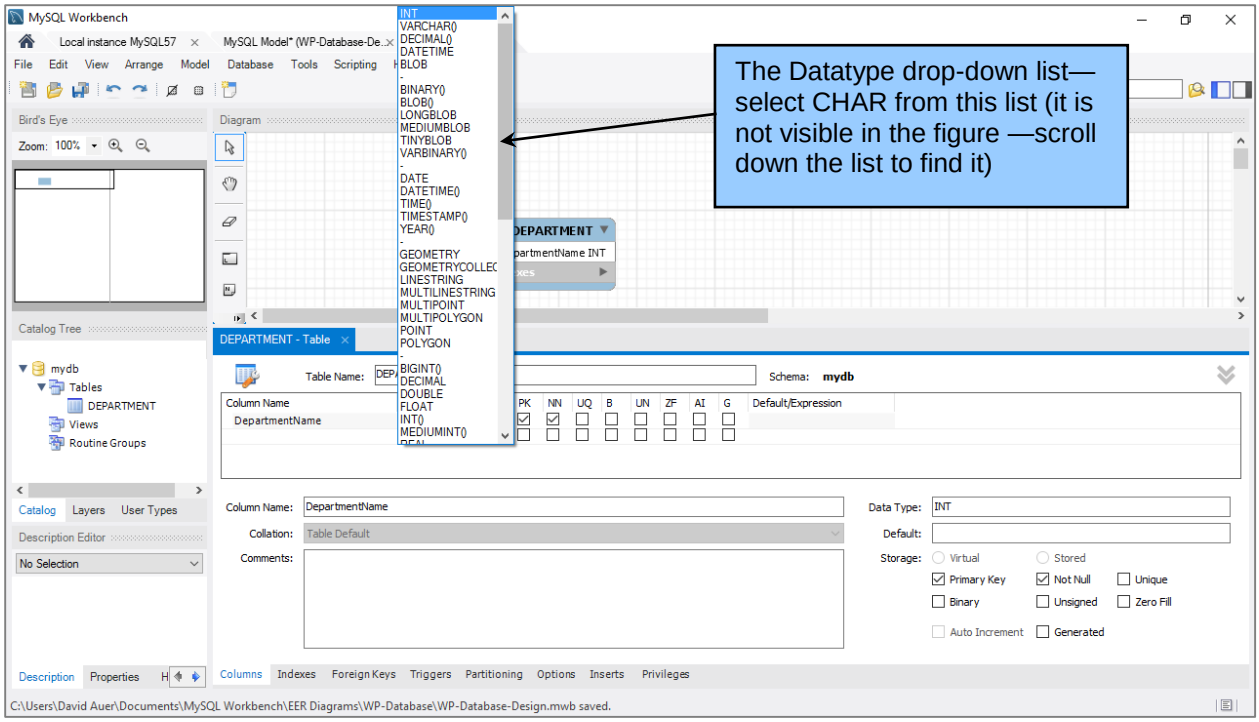


Figure C-56 — The Datatype Drop-Down List

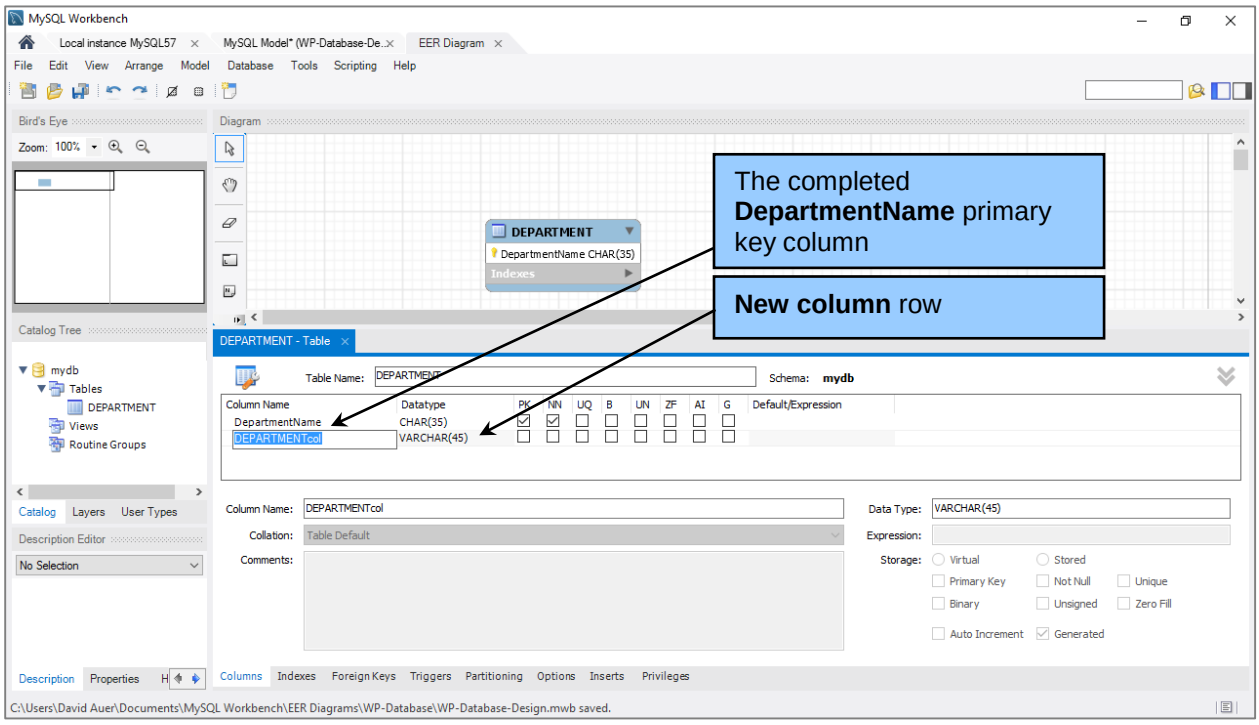


Figure C-57 — The Completed DepartmentName Primary Key Column

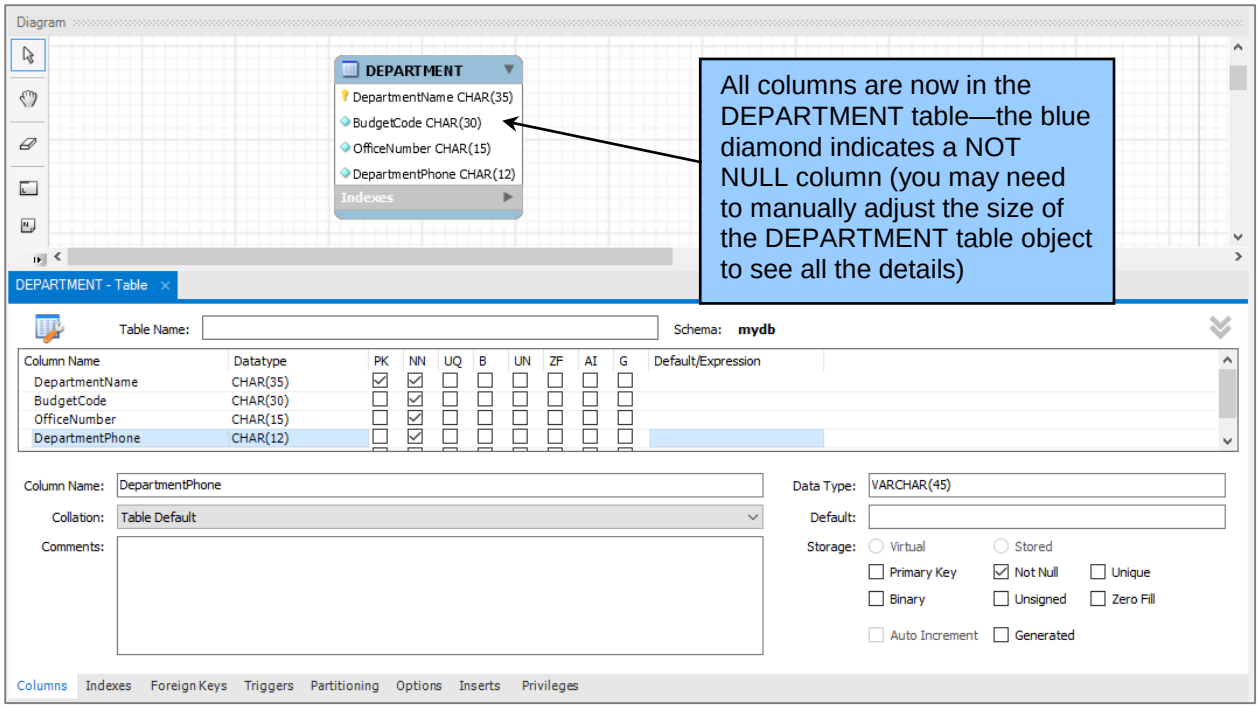


Figure C-58 — The Completed DEPARTMENT Table

Now, on your own, build the EMPLOYEE and PROJECT tables, but wait to build the ASSIGNMENT table until we discuss how to create relationships. The process is similar to the process we used to build the DEPARTMENT table, and the results are shown in Figure C-59. Note that the table objects have been resized and rearranged. After you have completed the three tables, close the MySQL Table Editor so your screen looks like Figure C-59. The blue diamonds in each table design indicate that NN is set, the white diamonds indicate NN is not set. You may find it easier to enter the column name, press tab, then enter the datatype directly for each column, then update the column characteristics boxes separately. To delete a column, right-click it and use **Delete Selected**.

Now that we have created the tables, we need to connect them with relationships. As shown in Figure C-60, the MySQL Workbench has buttons (labeled in IE Crow's Foot notation) to create a variety of relationships:

- **1:1 Non-identifying Relationship**—Used between two strong entities.
- **1:N Non-identifying Relationship**—Used between two strong entities.
- **1:1 Identifying Relationship**—Used between a strong entity and an ID-dependent weak entity—see discussion below.
- **1:N Identifying Relationship**—Used between a strong entity and an ID-dependent weak entity.
- **N:M Identifying Relationship**—Used between two strong entities—see discussion below.
- **Place a Relationship Using Existing Columns**—See discussion below.

The usage of **1:1 non-identifying**, **1:N non-identifying**, and **1:N identifying** relationships are standard and correct. However, by definition an identifying relationship has to be used in a 1:N relationship (see Chapter 4), so the **1:1** identifying relationship does not make sense. Similarly, pure N:M relationships only exist in data models, and they are always *non-identifying* relationships between two strong entities, so this symbol uses a dashed line instead of a solid one. However, the MySQL Workbench lets us edit relationships after we've created them so we can change any parameter we want to change.

By The Way

The MySQL Workbench uses the term **non-identifying** relationship, whereas in *Database Concepts* we use the term **nonidentifying** relationship. We have seen the term **non identifying** relationship used in other contexts. All three terms mean exactly the same thing, and which is used is a matter of style. Since MySQL Workbench uses *non-identifying*, we will also use that term in this appendix for consistency with the MySQL Workbench screenshots, while remaining well aware that we have used *nonidentfying* in *Database Concepts* itself.

Among the buttons that MySQL Workbench provides for creating relationships in the database design, the **Place a Relationship Using Existing Columns** choice is very useful. Normally, when we create a relationship, the MySQL Workbench automatically adds a foreign key, even if the column that should be the foreign key is already there! And this relationship usually turns out to be a non-identifying relation, despite the solid line shown on the button. Since both EMPLOYEE and PROJECT contain the column that

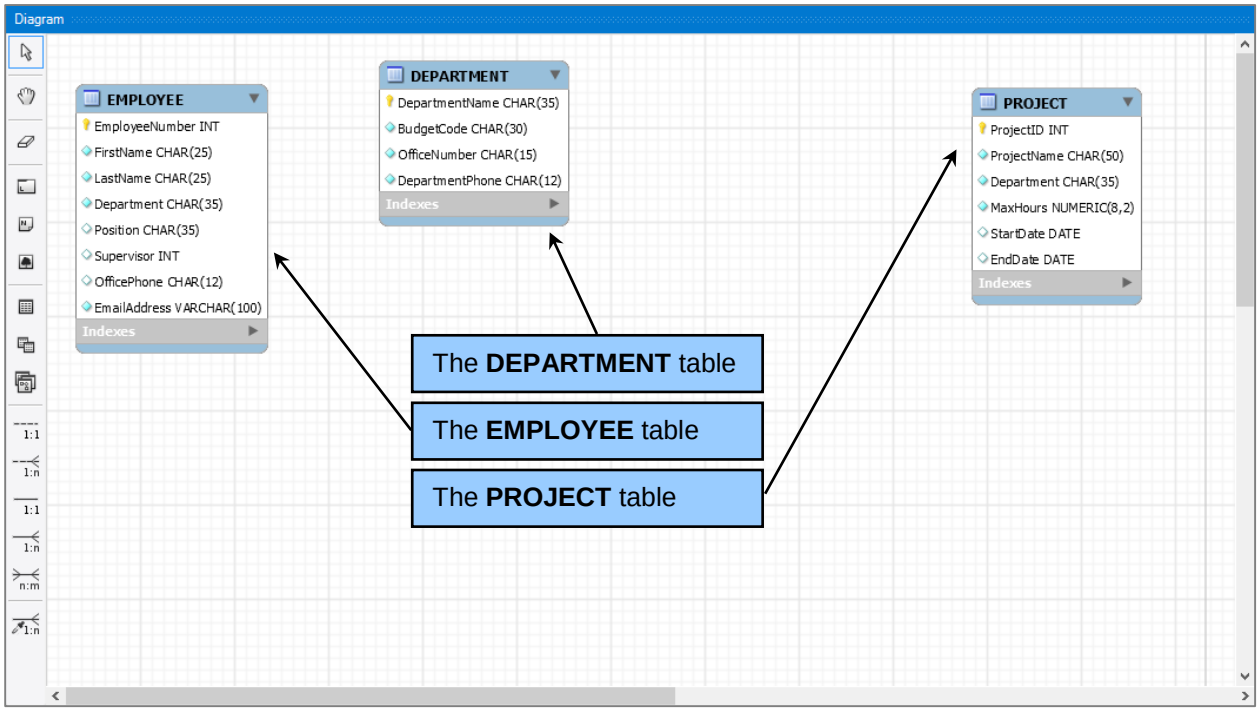


Figure C-59 — The Completed DEPARTMENT, EMPLOYEE, and PROJECT Tables

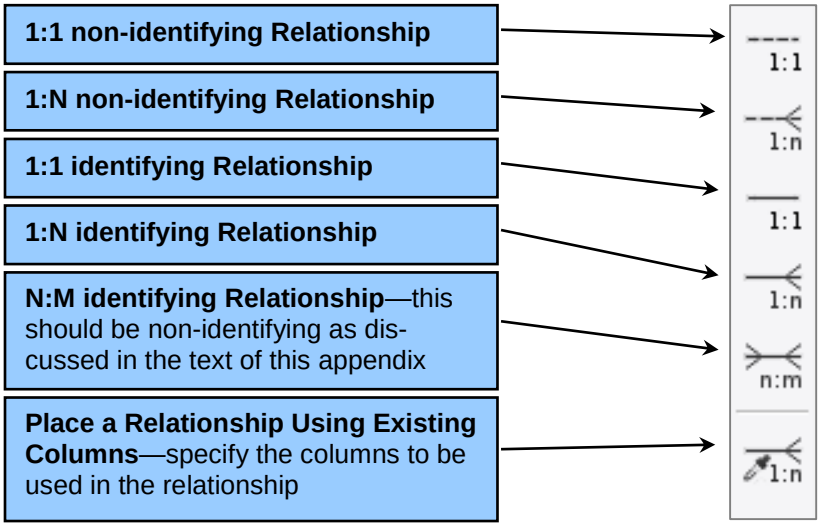


Figure C-60 — Relationships in the MySQL Workbench EER Diagram Toolbar

will be the foreign key—in both cases it is the Department column—we will use this button to create the relationships between the tables currently in the E-R diagram. We will start with the relationship between DEPARTMENT and EMPLOYEE.

Creating a 1:N Nonidentifying Relationship Between Two Tables:

1. Click the **Place a Relationship Using Existing Columns** button (see Figure C-61).
2. MySQL Workbench displays a Foreign Key Columns dialog box instructing us to *Pick one or more columns for the foreign key*, as shown in Figure C-61.
3. Click the **Department** column in EMPLOYEE to select it.
4. Click the **Pick Referenced Columns** button in the Foreign Key Columns dialog box, as shown in Figure C-62.
5. The Foreign Key Columns dialog box becomes the Referenced Columns dialog box, instructing us to *Pick matching columns for the referenced table*, as shown in Figure C-63.
6. Click the **DepartmentName** column in DEPARTMENT. A 1:N non-identifying relationship is created between DEPARTMENT (parent-the 1 side of the relationship) and EMPLOYEE (child-the N side of the relationship), as shown in Figure C-64. Note the red diamond indicating a foreign key in EMPLOYEE.
7. To see the details of the relationship, right-click the relationship line, then click Edit Relationship, and then click the **Foreign Key tab** at the bottom of the Relationship pane. The relationship parameters are displayed as shown in Figure C-65.

The MySQL Relationship Editor shown in Figure C-65 shows this relationship (it has been named fk_EMPLOYEE_DEPARTMENT by the MySQL Workbench, but we can change the name if desired), and can control the name in the SQL code when we actually construct the tables and relationships in our DBMS (in the SQL CREATE TABLE EMPLOYEE statement in Figure C-16 we used a FOREIGN KEY CONSTRAINT phrase to name this corresponding foreign key constraint as EMP_DEPART_FK).

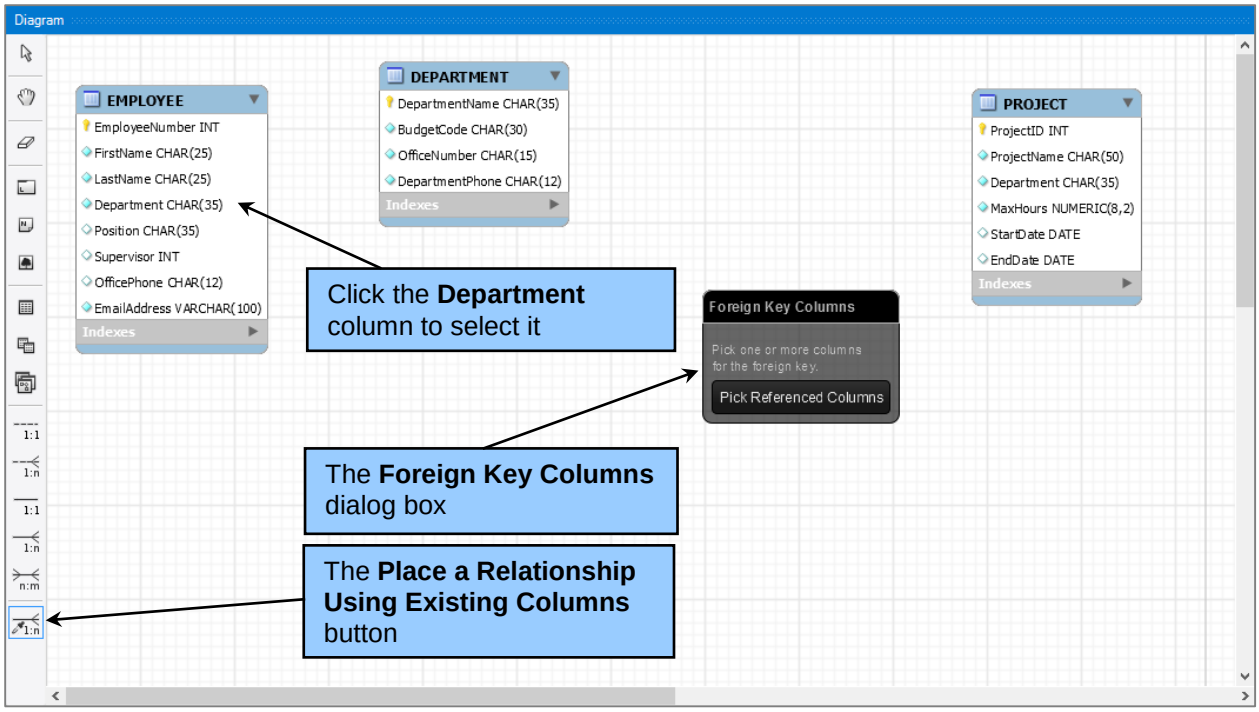


Figure C-61 — The Foreign Key Columns Dialog Box – Pick Columns for Foreign Key

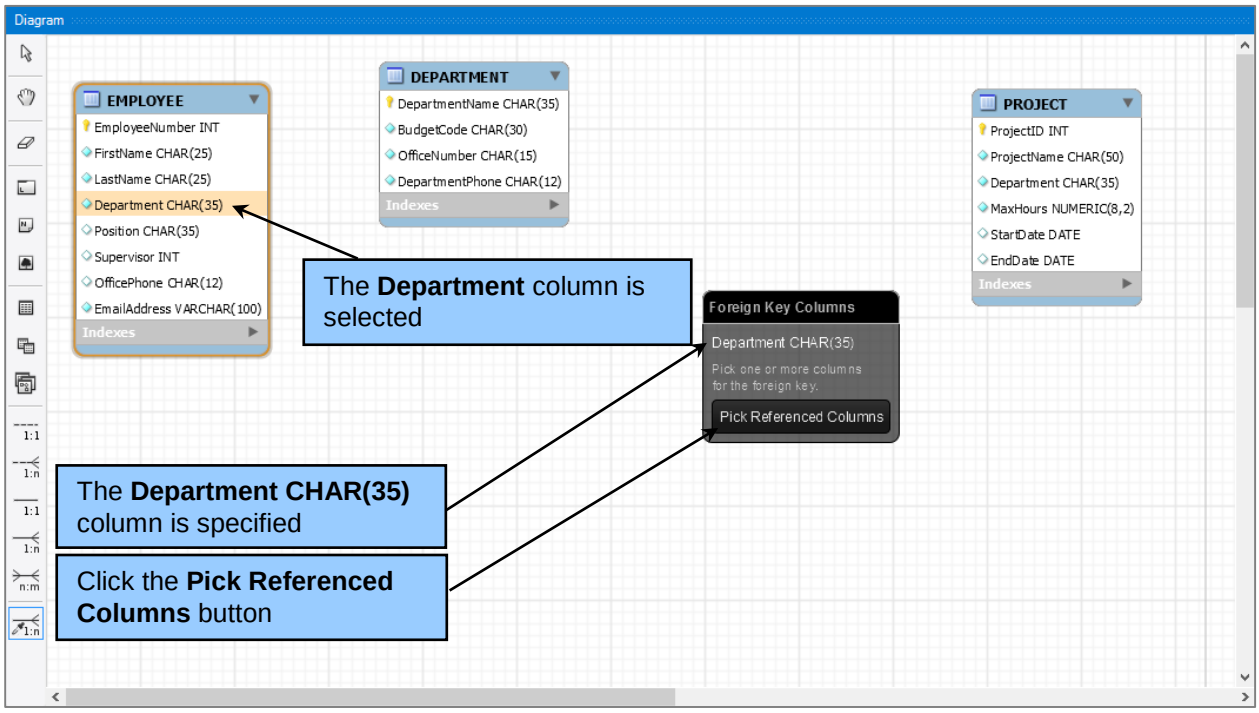


Figure C-62 — EMPLOYEE.Department Selected as the Foreign Key Column

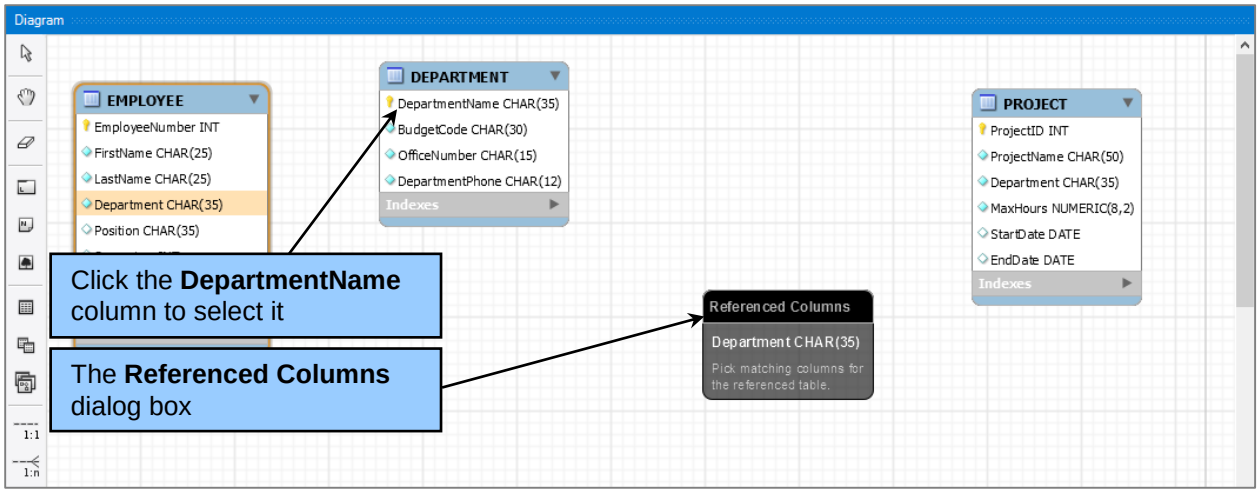


Figure C-63— The Referenced Columns Dialog Box – Pick Referenced Primary Key

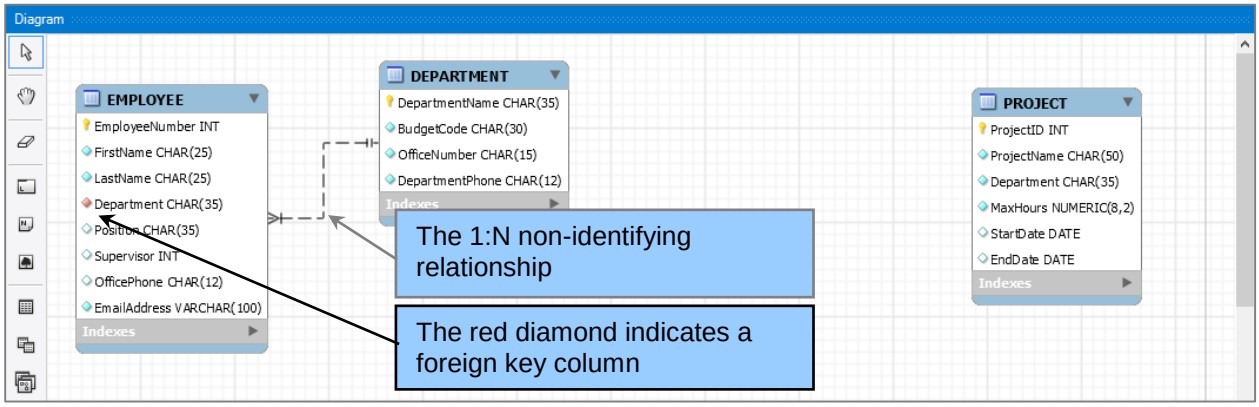


Figure C-64 — The Completed 1:N Non-identifying Relationship

The options on the Foreign Key page shown in Figure C-65 allow us to control all aspects of the relationship. As shown there, the relationship is one-to-many (1:N), non-identifying, with both DEPARTMENT and EMPLOYEE having mandatory participation in the relationship (i.e., minimum cardinality of 1). The non-identifying 1:N parameters are correct, but what about the minimum cardinalities?

Does a DEPARTMENT have to have at least one employee? This is actually a business rule question, but we will assume that the answer for WP is yes, meaning that WP does not allow departments without employees to exist.

Does an EMPLOYEE have to be assigned to a department? Again, this is a business rule question, but the fact that EMPLOYEE.Department is NOT NULL with a DEFAULT value of Human Resources is a good indication that the answer for WP is yes, meaning that WP does not allow employees unassigned to departments to exist.

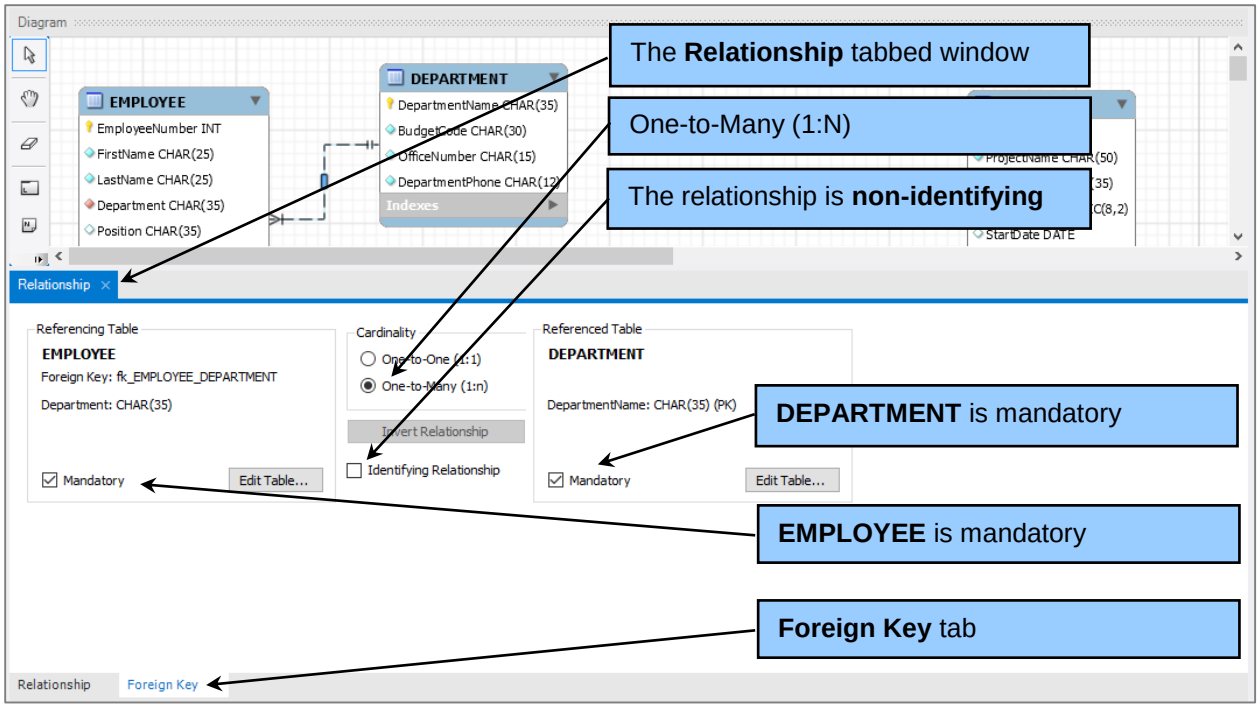


Figure C-65 — The Foreign Key Properties

Therefore, the mandatory parameters are correct, and our relationship is correct as drawn.

8. Close the MySQL Relationship Editor by click the **Close [X]** button on the tab.

On your own, create the relationship between DEPARTMENT and PROJECT the same way. We will assume that every PROJECT must have a sponsoring DEPARTMENT but that a DEPARTMENT is not required to have any PROJECTs in the works. Connect the FK Department in the PROJECT table to the PK DepartmentName in the DEPARTMENT table. Finally, we need to add the recursive relationship between EMPLOYEE.Supervisor and EMPLOYEE.EmployeeNumber. Connect the FK Supervisor in the EMPLOYEE table to the PK EmployeeNumber in the EMPLOYEE table. Note that the participation is optional, so edit the recursive relationship, select the Foreign Key tab, and de-select the Mandatory boxes for both sides. The completed E-R diagram with the additional relationships is shown in Figure C-66.

At this point we still need to build the ASSIGNMENT table and its relationships with EMPLOYEE and PROJECT. However, since ASSIGNMENT is an association table in an associations relationship with EMPLOYEE and PROJECT (association relationships are discussed in Chapter 5) and since association tables are just intersection tables with additional attributes (again, as discussed in Chapter 5), we will use this as an opportunity to illustrate how MySQL Workbench handles N:M relationships.

As discussed in Chapters 4 and 5, an N:M relationship only exists in a data model (as a non-identifying relationship between two strong entities). In a database design, the N:M relationship becomes two 1:N ID-dependent identifying relationships linking the two original tables through a new, third table called an intersection table. However, MySQL Workbench only builds database designs and will automatically create the intersection table with the two 1:N relationships whenever we specify an N:M relationship.

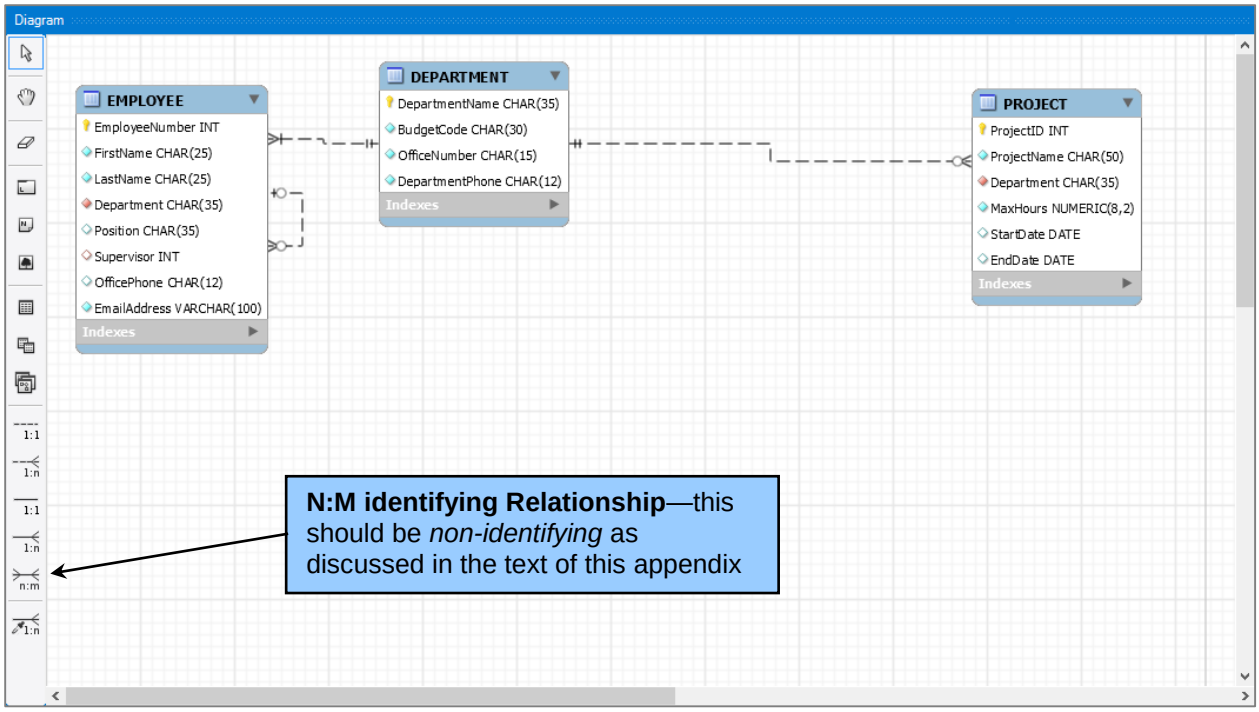


Figure C-66 — The Completed DEPARTMENT, EMPLOYEE and PROJECT Relationships

Creating an N:M Nonidentifying Relationship Between Two Tables:

1. Click the **N:M identifying Relationship** button, then click the **EMPLOYEE** table, and then click the **PROJECT** table.
2. As shown in Figure C-67, the MySQL Workbench creates an *intersection table* named EMPLOYEE_has_PROJECT and places it in the E-R diagram together with two 1:N ID-dependent identifying relationships between (1) EMPLOYEE_has_PROJECT and EMPLOYEE and (2) EMPLOYEE_has_PROJECT and PROJECT.

We have just demonstrated that the MySQL Workbench can only create database designs, not data models, by showing how MySQL Workbench automatically converts a M:N relationship into two 1:N relationships to an intersection table . Now we simply have to do some editing in the MySQL Table Editor by double-clicking the EMPLOYEE_has_PROJECT table:

- Change the table name to ASSIGNMENT.
- Change the primary key attribute names to EmployeeNumber and ProjectID.
- Add the HoursWorked attribute as NUMERIC(6,2).

The completed diagram is shown in Figure C-68. What about the relationship cardinalities? We will assume that every project has to have at least one employee assigned to it, and that every employee has to work on at least one project. Therefore, the minimum cardinalities are correct as created. The final E-R diagram is shown in Figures C-67 and C-68 and, except for the exact arrangement of the tables and relationships, is the same as the WP database design shown in Figures 5-16 and 5-17. And that completes our introduction to MySQL.

3. Save the WP-Database-Design in the MySQL Workbench, and close the EER Diagram window (the X in the EER Diagram tab).
4. Close the MySQL Model window (the X in the MySQL Model tab).
5. Close the MySQL Workbench.

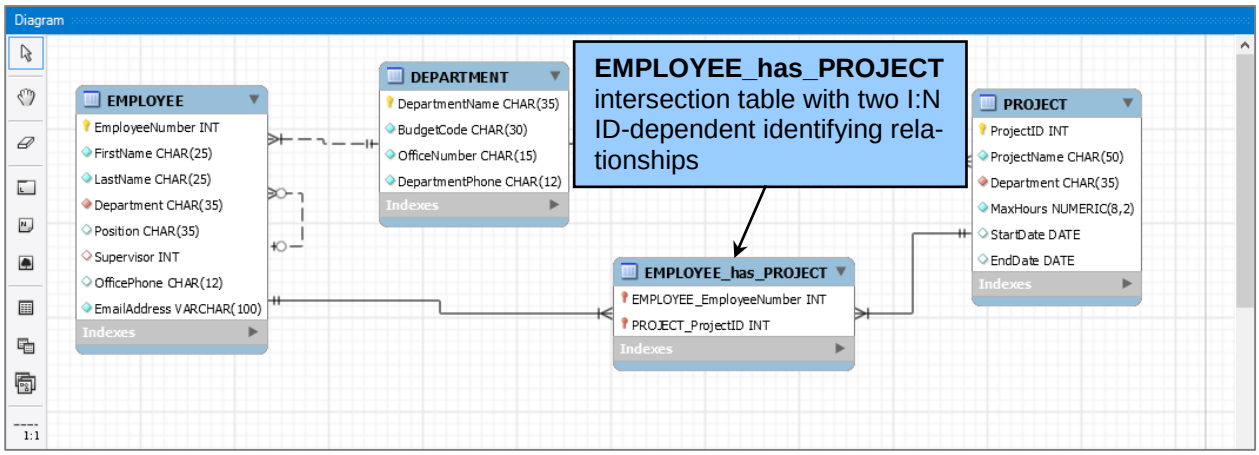


Figure C-67 — The EMPLOYEE_has_PROJECT Intersection Table

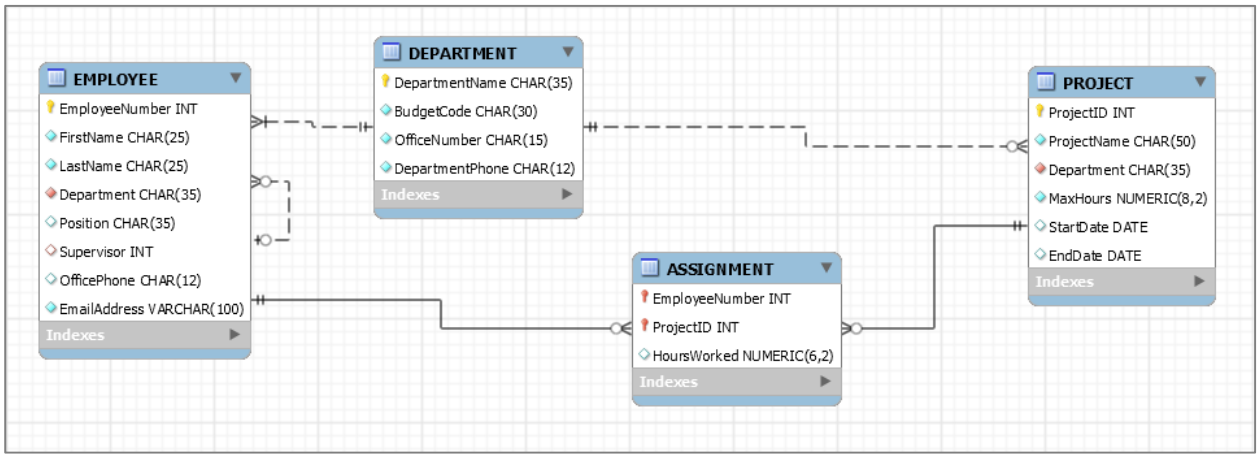


Figure C-68 — The Completed E-R Diagram

KEY TERMS

***.sql**	**Connect to Database dialog box**
Connect to MySQL Server dialog box	**data model**
database design	**default schema**
Execute SQL Script in Connected Server button	**Execute Current SQL Statement in Connected Server button**
MySQL Connector/ODBC	**MySQL Community Server**
MySQL Workbench	**Object browser**
Open Connection to Start Querying	**Place a New Table button**
Place a Relationship Using Existing Columns button	**Result(1) window**
Root user account	**schema**
SQL script	

REVIEW QUESTIONS

C.1 What is MySQL?

C.2 What is the primary advantage of using MySQL instead of SQL Server 2016?

C.3 What are the four MySQL programs that are recommended as a useful set of MySQL software products?

C.4 What are two purposes of the MySQL Workbench?

C.5 What is a MySQL schema?

C.6 How do you create a new database in MySQL?

C.7 What is a MySQL default schema? How do you specify the default schema?

C.8 What is an SQL script? What types of SQL statements and commands can be run more efficiently as scripts?

C.9 What tool(s) can you use to create an SQL script?

C.10 What file extension should you use with MySQL scripts?

C.11 How do you create, save, and run an SQL script in MySQL?

C.12 How does MySQL use the AUTO_INCREMENT keyword? What limitations are there on the use of the AUTO_INCREMENT keyword?

C.13 How do you create and run an SQL query in MySQL?

C.14 How do you obtain online help when using the MySQL Workbench? Where is the help information displayed?

C.15 What is the purpose of the MySQL Connector/OBDC? How do you install it, and where does it appear after it is installed?

C.16. How do you import Microsoft Excel 2016 into a MySQL 5.7 table?

C.17. How do you create user accounts in MySQL 5.7? How do you give them appropriate permissions to a specific database?

C.18. Why would you want to create an ODBC connection to link a Microsoft Access 2016 to a MySQL 5.7 Database?

C.19. What is an ODBC DSN? Why is one needed?

C.20. How do you create an ODBC connection to link a Microsoft Access 2016 database to a MySQL 5.7 Database?

C.21 The MySQL Workbench creates *database models*. What is a data model? What is a database design? Which does MySQL Workbench create?

C.22 How do you create a database model in MySQL Workbench?

C.23 How do you create a new E-R diagram in the MySQL Workbench?

C.24 How do you create a table in an E-R diagram?

C.25 How do you create a 1:N non-identifying relationship in an E-R diagram?

C.26 How does the MySQL Workbench handle N:M relationships?

EXERCISES

C.27 If you haven't already done so, download and install MySQL 5.7 Community Server and the MySQL Workbench. Use the default setting for each installation. When you are asked for a password for the Root user account, create one.

C.28 If you haven't already done so, work through the steps described in this appendix to create and populate the WP database.

C.29 Use MySQL and the MySQL Workbench to create and run the following SQL queries in Chapter 3**,** starting on page 160.

SQL-Query-CH03-01 through SQL-Query-CH03-32

SQL-Query-CH03-35 through SQL-Query-CH03-56

Save each query as follows:

- Create and run each query in the MySQL Workbench.
- After you have run each query, use the **File | Save Script As** command to save the query. By default, MySQL saves each file as an *.sql script. Name your queries in numerical sequence, starting with the file name MySQL-SQL-Query-CH03-01.sql.

C.30 Use MySQL and the MySQL Workbench to run one or more of the saved SQL queries you created in exercise C.23:

- At this point, you should have only one SQL Script window open. Open a query with the **File | Open SQL Script** command.
- Experiment with opening and closing SQL Script windows and running various queries in these windows.

C.31 Complete exercise 3.59 using MySQL. Start each saved query name with MySQL- and use the default *.sql file extension. (The first saved query name should be MySQL-SQL-Query-AWE-3-1-A.sql.)

C.32 Complete exercise 3.60 using MySQL. Start the saved query name with MySQL- and use the default *.sql file extension. Create as many scripts as you need for Parts A through D. The saved query name should be MySQL-SQL-Query-AWE-3-3-E.sql.

C.33 If you have not already done so, import the **COMPUTER** table from Microsoft Excel into the MySQL WP database as explained in the text.

Database Column Characteristics for the WP COMPUTER_ASSIGNMENT Table

Column Name	Type	Key	Required	Remarks
SerialNumber	Number	Primary Key, Foreign Key	Yes	Long Integer
EmployeeNumber	Number	Primary Key, Foreign Key	Yes	Long Integer
DateAssigned	Date/Time	Primary Key	Yes	Medium Date

Figure C-69 — Database Column Characteristics for the WP COMPUTER_ASSIGNMENT Table

C.34 Import the **COMPUTER_ASSIGNMENT** table from Microsoft Excel into the MySQL WP database. Column characteristics are shown in Figure C-69 (this table shows the column characteristics with Microsoft Access 2016 data types). How should this table be linked to the EMPLOYEE table and the COMPUTER table by foreign keys? Be sure to include these foreign keys in your final COMPUTER_ASSIGNMENT table structure when you create it in MySQL.

C.35 If you have not already done so, create the **WP-User** user account and associated permissions in the MySQL WP database as explained in the text.

C.36 Create a user account in the MySQL Server WP database named **WP-Reader**. Give this user **SQL Server authentication** with the password of **WP-Reader+password** and with other password settings to match those shown in Figure C-36. Give WP-Reader a only SELECT object rights to the WP database.

C.37 Create a Microsoft Access 2016 database named **WPIS_RO.accdb** where RO stands for "read-only." This database will be a *read-only* application for the MySQL WP database, which will allow users to read and query the data in the WP database but not to make any updates to the data or to insert new data. Then:

1. Set the WPIS_RO.accdb database to use **SQL Server Compatible Syntax (ANSI 92)**.
2. Link the WPIS_RO.accdb database to the MySQL WP database. When you create your File Data Source DSN, name the DSN **WP_RO**, and use the WP-Reader user account (as detailed in Exercise A.33) for SQL Server authentication.
3. Import all existing tables (including the COMPUTER and COMPUTER_ASSIGNMENT tables if they have been imported as detailed in Exercises A.30 and A.31).
4. Create a form to show all the data in PROJECT table named **WP Projects Form**.
5. Create a report to show all the data in the PROJECT table named **WP Projects Report**.

C.38 The database design for the Wallingford Motor Customer Relationship Management database is shown in Figure AW-5-1. In the MySQL Workbench, create a new MySQL Database Model named WMCRM-Database-Design. In this model, add an E-R diagram, and recreate the database design shown in Figure AW-5-1 in this diagram.

C.39 The database design for Heather Sweeney Designs is shown in Figure 5-27. In the MySQL Workbench, create a new MySQL Database Model named HSD-Database-Design. In this model, add an E-R diagram, and recreate the database design shown in Figure 5-27 in this diagram.