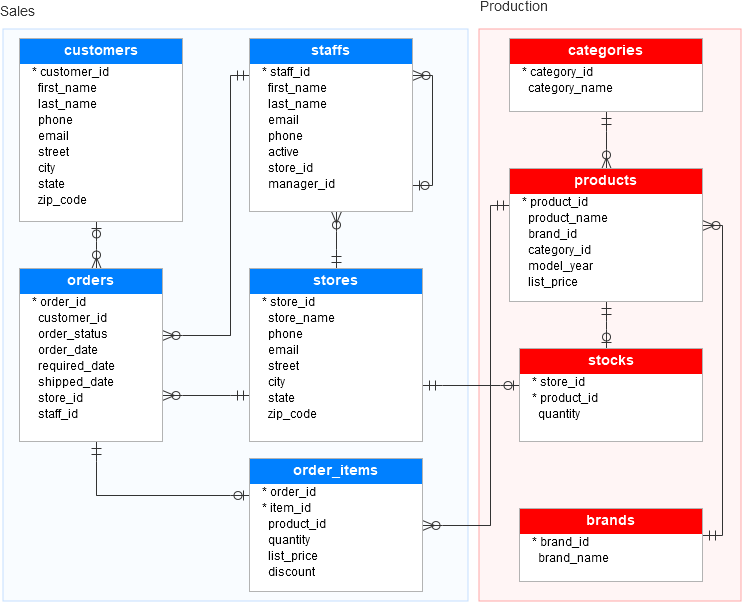
The following illustrates the BikeStores database diagram:



As you can see from the diagram, the BikeStores sample database has two schemas sales and production, and these schemas have nine tables.

Database Tables

Table sales.stores

The  sales.stores table includes the store’s information. Each store has a store name, contact information such as phone and email, and an address including street, city, state, and zip code.

**CREATE** **TABLE** sales.stores (

store\_id INT **IDENTITY** (1, 1) PRIMARY **KEY**,

store\_name VARCHAR (255) **NOT** NULL,

phone VARCHAR (25),

email VARCHAR (255),

street VARCHAR (255),

city VARCHAR (255),

state VARCHAR (10),

zip\_code VARCHAR (5)

);

Table sales.staffs

The  sales.staffs table stores the essential information of staffs including first name, last name. It also contains the communication information such as email and phone.

A staff works at a store specified by the value in the store\_id column. A store can have one or more staffs.

A staff reports to a store manager specified by the value in the manager\_id column. If the value in the manager\_id is null, then the staff is the top manager.

If a staff no longer works for any stores, the value in the active column is set to zero.

**CREATE** **TABLE** sales.staffs (

staff\_id INT **IDENTITY** (1, 1) PRIMARY **KEY**,

first\_name VARCHAR (50) **NOT** NULL,

last\_name VARCHAR (50) **NOT** NULL,

email VARCHAR (255) **NOT** NULL **UNIQUE**,

phone VARCHAR (25),

active tinyint **NOT** NULL,

store\_id INT **NOT** NULL,

manager\_id INT,

**FOREIGN** **KEY** (store\_id)

**REFERENCES** sales.stores (store\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**,

**FOREIGN** **KEY** (manager\_id)

**REFERENCES** sales.staffs (staff\_id)

**ON** **DELETE** **NO** **ACTION** **ON** **UPDATE** **NO** **ACTION**

);

Table production.categories

The production.categories table stores the bike’s categories such as children bicycles, comfort bicycles, and electric bikes.

**CREATE** **TABLE** production.categories (

category\_id INT **IDENTITY** (1, 1) PRIMARY **KEY**,

category\_name VARCHAR (255) **NOT** NULL

);

Table production.brands

The  production.brands table stores the brand’s information of bikes, for example, Electra, Haro, and Heller.

**CREATE** **TABLE** production.brands (

brand\_id INT **IDENTITY** (1, 1) PRIMARY **KEY**,

brand\_name VARCHAR (255) **NOT** NULL

);

Table production.products

The production.products table stores the product’s information such as name, brand, category, model year, and list price.

Each product belongs to a brand specified by the brand\_id column. Hence, a brand may have zero or many products.

Each product also belongs a category specified by the category\_id column. Also, each category may have zero or many products.

**CREATE** **TABLE** production.products (

product\_id INT **IDENTITY** (1, 1) PRIMARY **KEY**,

product\_name VARCHAR (255) **NOT** NULL,

brand\_id INT **NOT** NULL,

category\_id INT **NOT** NULL,

model\_year SMALLINT **NOT** NULL,

list\_price DECIMAL (10, 2) **NOT** NULL,

**FOREIGN** **KEY** (category\_id)

**REFERENCES** production.categories (category\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**,

**FOREIGN** **KEY** (brand\_id)

**REFERENCES** sales.brands (brand\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**

);

Table sales.customers

The  sales.customers table stores customer’s information including first name, last name, phone, email, street, city, state and zip code.

**CREATE** **TABLE** sales.customers (

customer\_id INT **IDENTITY** (1, 1) PRIMARY **KEY**,

first\_name VARCHAR (255) **NOT** NULL,

last\_name VARCHAR (255) **NOT** NULL,

phone VARCHAR (25),

email VARCHAR (255) **NOT** NULL,

street VARCHAR (255),

city VARCHAR (50),

state VARCHAR (25),

zip\_code VARCHAR (5)

);

Table sales.orders

The sales.orders table stores the sales order’s header information including customer, order status, order date, required date, shipped date.

It also stores the information on where the sales transaction created (store) and who created it (staff).

Each sales order has a row in the sales\_orders table. A sales order has one or many line items stored in the sales.order\_items table.

**CREATE** **TABLE** sales.orders (

order\_id INT **IDENTITY** (1, 1) PRIMARY **KEY**,

customer\_id INT,

order\_status tinyint **NOT** NULL,

*-- Order status: 1 = Pending; 2 = Processing; 3 = Rejected; 4 = Completed*

order\_date DATE **NOT** NULL,

required\_date DATE **NOT** NULL,

shipped\_date DATE,

store\_id INT **NOT** NULL,

staff\_id INT **NOT** NULL,

**FOREIGN** **KEY** (customer\_id)

**REFERENCES** sales.customers (customer\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**,

**FOREIGN** **KEY** (store\_id)

**REFERENCES** sales.stores (store\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**,

**FOREIGN** **KEY** (staff\_id)

**REFERENCES** sales.staffs (staff\_id)

**ON** **DELETE** **NO** **ACTION** **ON** **UPDATE** **NO** **ACTION**

);

Table sales.order\_items

The sales.order\_items table stores the line items of a sales order. Each line item belongs to a sales order specified by the order\_id column.

A sales order line item includes product, order quantity, list price and discount.

**CREATE** **TABLE** sales.order\_items(

order\_id INT,

item\_id INT,

product\_id INT **NOT** NULL,

quantity INT **NOT** NULL,

list\_price DECIMAL (10, 2) **NOT** NULL,

discount DECIMAL (4, 2) **NOT** NULL **DEFAULT** 0,

PRIMARY **KEY** (order\_id, item\_id),

**FOREIGN** **KEY** (order\_id)

**REFERENCES** sales.orders (order\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**,

**FOREIGN** **KEY** (product\_id)

**REFERENCES** production.products (product\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**

);

Table production.stocks

The production.stocks table stores the inventory information i.e. the quantity of a particular product in a specific store.

**CREATE** **TABLE** production.stocks (

store\_id INT,

product\_id INT,

quantity INT,

PRIMARY **KEY** (store\_id, product\_id),

**FOREIGN** **KEY** (store\_id)

**REFERENCES** sales.stores (store\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**,

**FOREIGN** **KEY** (product\_id)

**REFERENCES** production.products (product\_id)

**ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**

);

First, you need to download the following zip file if you have not done so:

[Download SQL Server Sample Database](https://cdn.sqlservertutorial.net/wp-content/uploads/SQL-Server-Sample-Database.zip)

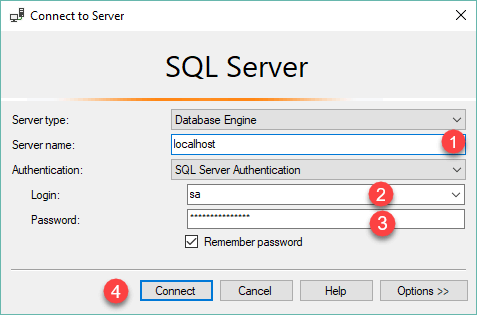
Second, uncompress the zip file, you will see three SQL script files:

* BikeStores Sample Database - create objects.sql – this file is for creating database objects including schemas and tables.
* BikeStores Sample Database - load data.sql – this file is for inserting data into the tables
* BikeStores Sample Database - drop all objects.sql – this file is for removing the tables and their schemas from the sample database. It is useful when you want to refresh the sample database.

Third, let’s create a database, create the schemas and tables, and load the sample data.

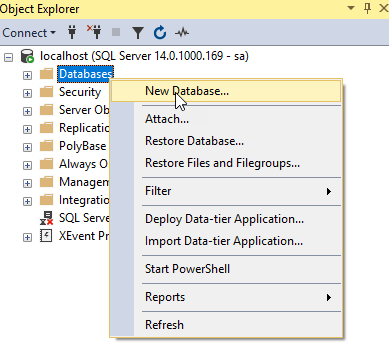
### Step 1

Connect to the SQL Server by (1) choosing the server name, (2) enter the user and (3) password and (4) click the **Connect** button.



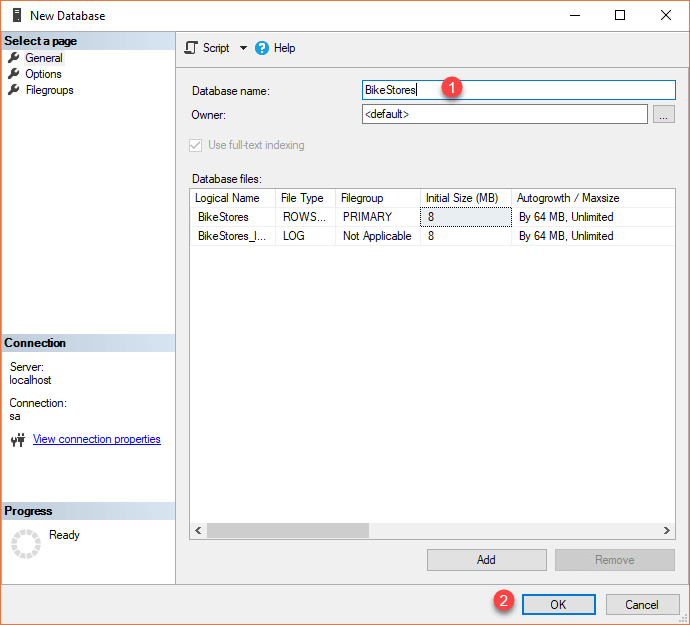
### Step 2

Right-click the **Databases** node in the **Object Explorer** and select the **New Database…** menu item



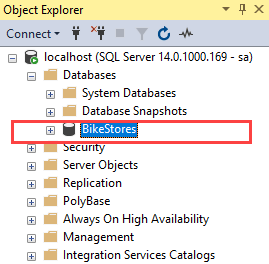
### Step 3

(1) Enter the **Database name**as BikeStores and (2) click the **OK** button to create the new database.



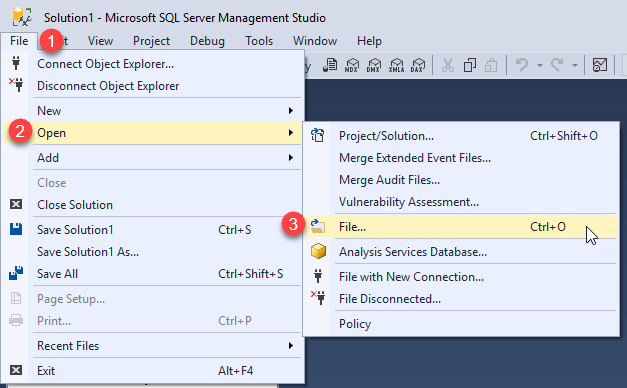
### Step 4

If everything is fine, you will see the database **BikeStores** appears under Databases node as shown in the screenshot below:



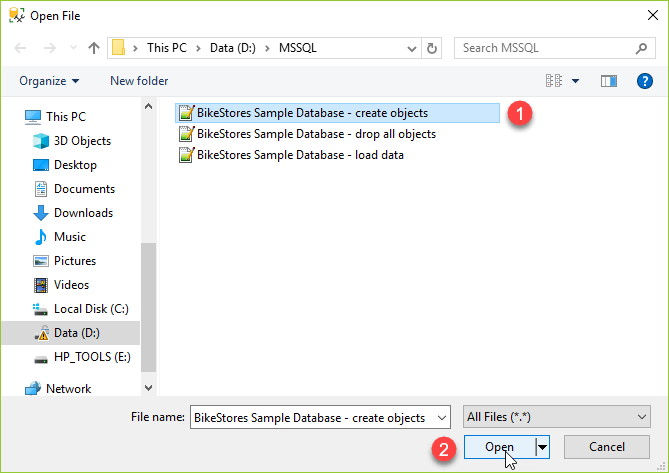
### Step 5

From the File menu, choose Open > File… menu item to open a script file.



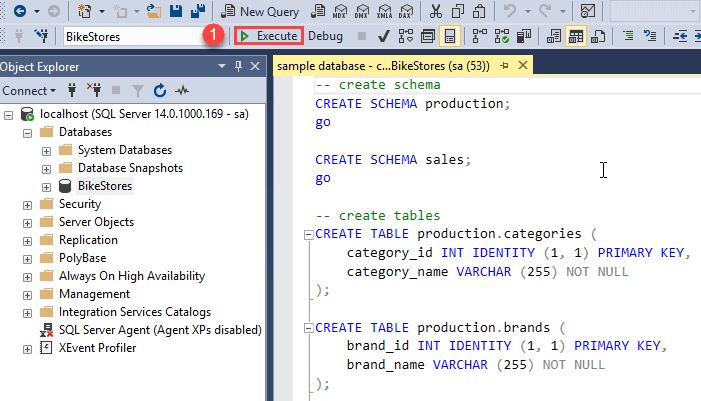
### Step 6

Select the **BikeStores Sample Database – create**objects.sql file and click the Open button

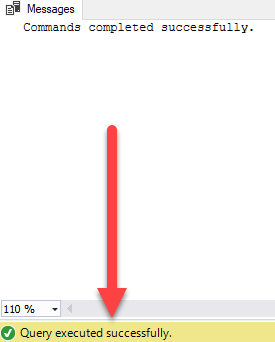


### Step 7

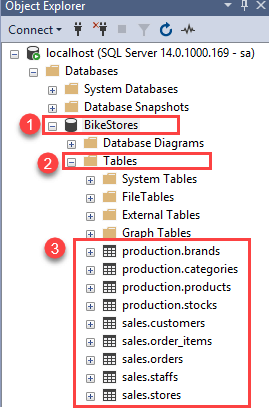
Click the **Execute** button to execute the SQL script.



You should see the following result indicated that the query executed successfully.

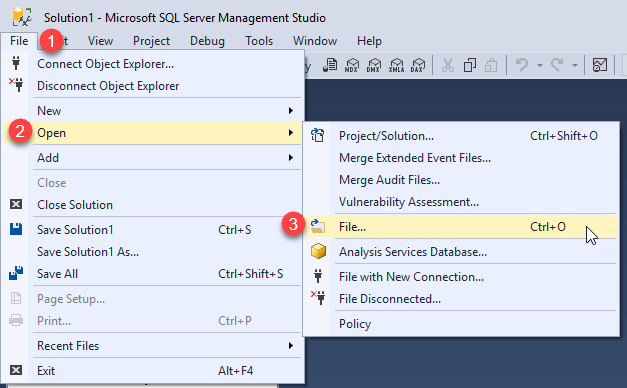


If you expand the **BikeStores > Tables**, you will see the schemas and their tables are created as shown below:



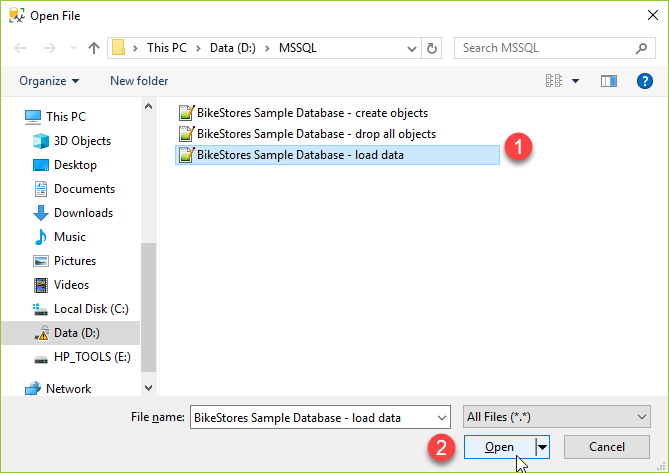
### Step 8

Open the file for loading data into the tables.



### Step 9

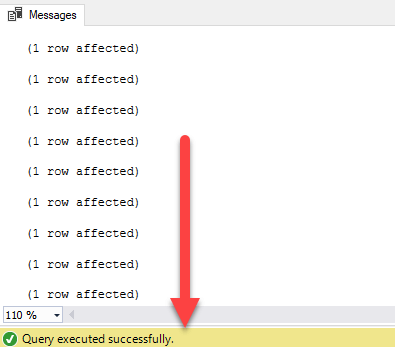
Choose the **BikeStores Sample Database – load data.sql** file and click the Open button.



### Step 10

Click the **Execute** button to load data into the tables.

You should see the following message indicating that all the statements in the script were executed successfully.



In this