# Introduction to MySQL

# Outline

- How to Manage your data?
- Introduction to MySQL
- Connecting and Disconnecting
- Entering Basic Queries
- Creating and Using a Database

# How to Manage your data?

- Access vs. Excel
- In both Access and Excel, you can:
  - ✓Import data from external sources
  - ✓Sort and filter your data.
  - ✓Do calculations to derive the information you want.
- Excel is spreadsheet software, that stores data in columns and rows.
- Access stores data in tables similar to worksheets in Excel

Source: http://office.microsoft.com/en-us/access-help/using-access-or-excel-to-manage-your-data-HA001042918.aspx

# How to Manage your data?

- However, table in Access are designed for complex querying in relation to data stored in other tables.

**both programs work well for managing data**

Source: http://office.microsoft.com/en-us/access-help/using-access-or-excel-to-manage-your-data-HA001042918.aspx

# How to Manage your data?

- The key to your decision:

  *Is the data relational or not?*

- **Flat** or **non-relational** data:
  - ✓ data in each column is directly related and need only reside in a single, flat table.
- **Relational** data:
  - ✓ Data needs to be stored in more than one table
  - ✓ Each table is a description of a type of data
  - ✓ For example: customer information in one table and their orders in another table

Source: http://office.microsoft.com/en-us/access-help/using-access-or-excel-to-manage-your-data-HA001042918.aspx

# How to Manage your data?

- Use Access when you:
  - ✓Require a relational database (multiple tables) to store your data.
  - ✓May need to add more tables in the future to an originally flat or non-relational data set.
  - ✓Want to run **complex queries**.
- Use Excel when you:
  - ✓Require a flat or **non-relational** view of your data
  - ✓Want to run primarily **calculations** and **statistical comparisons** on your data
  - ✓Know your **dataset is manageable in size**

Source: http://office.microsoft.com/en-us/access-help/using-access-or-excel-to-manage-your-data-HA001042918.aspx

# MySQL

- MySQL is a very popular, open source database (since 2008).
- MySQL is a ***relational*** database.
- Officially pronounced “my S-Q-L” (also my sequel).
- MySQL runs as a server and provides multi-user access to a number of databases
- Handles very large databases;  very fast performance.
- Why are we using MySQL?
  - Free (much cheaper than Oracle!)
  - Everyone can install MySQL locally.

# When to use MySQL

- MySQL is a much more stable and faster database than Microsoft Access
- Microsoft Access has **2GB** size limitations
- MySQL is free!

# Installing MySQL

- MySQL Community Server 5.6.11

The software can be downloaded from http://www.mysql.com/downloads/mysql/

- Mysql-connector-net-6.2

The software can be downloaded from http://www.mysql.com/downloads/connector/net/6.2.html#downloads

# User Interface

- MySQL Workbench
  - ✓Database Design & Modeling
  - ✓Database Administration
  - ✓Database Migration

The software can be downloaded from:

http://dev.mysql.com/downloads/tools/workbench/

# Download MySQL Community Server

MySQL Community Edition is a freely downloadable version of the world's most popular open source database that is
enthusiasts.

MySQL Cluster Community Edition is available as a separate download. The reason for this change is so that MySQL C
sources of MySQL Cluster Carrier Grade Edition.

Important Platform Support Updates

Generally Available (GA) Releases

## MySQL Community Server 5.6.10

Select Platform:

Select Platform...  Select

**Recommended Download:**

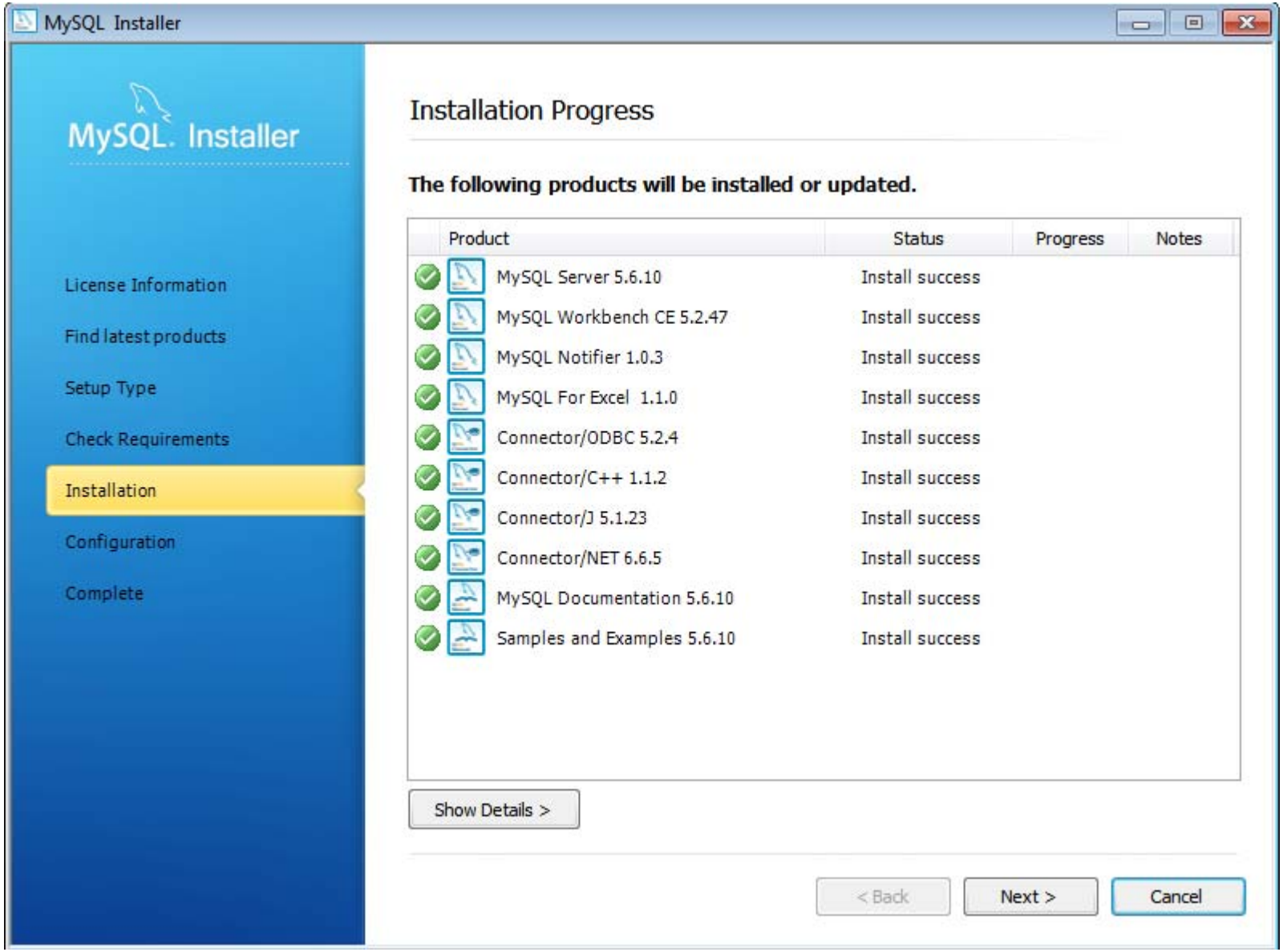
MySQL Installer 5.6
for Windows

**All MySQL Products. For All Windows Platforms.
In One Package.**

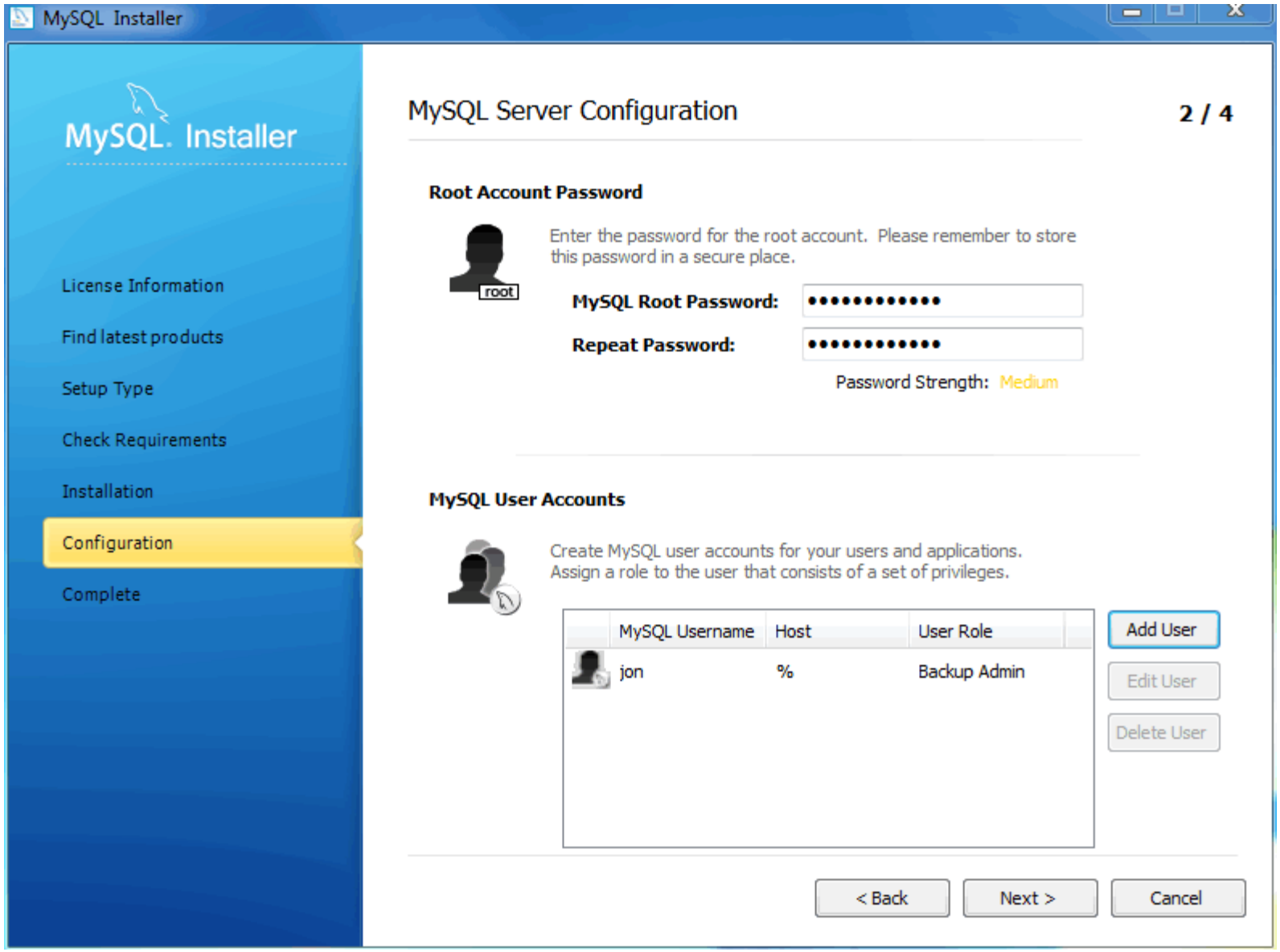
Starting with MySQL 5.6 the MySQL Installer package replaces the server-only MSI packages.

Download

http://dev.mysql.com/doc/refman/5.6/en/mysql-installer-gui.html

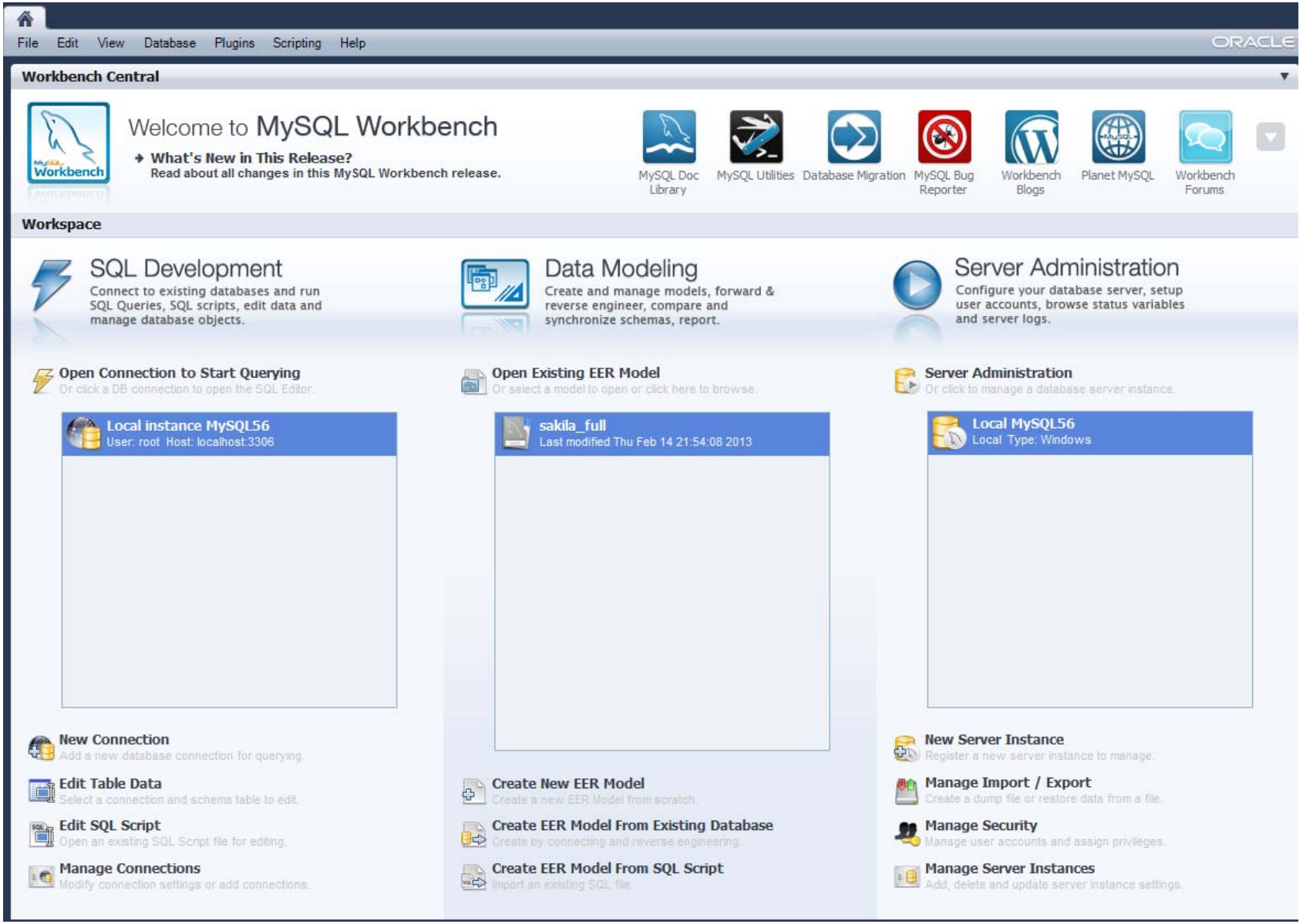



http://dev.mysql.com/doc/refman/5.6/en/mysql-installer-gui.html

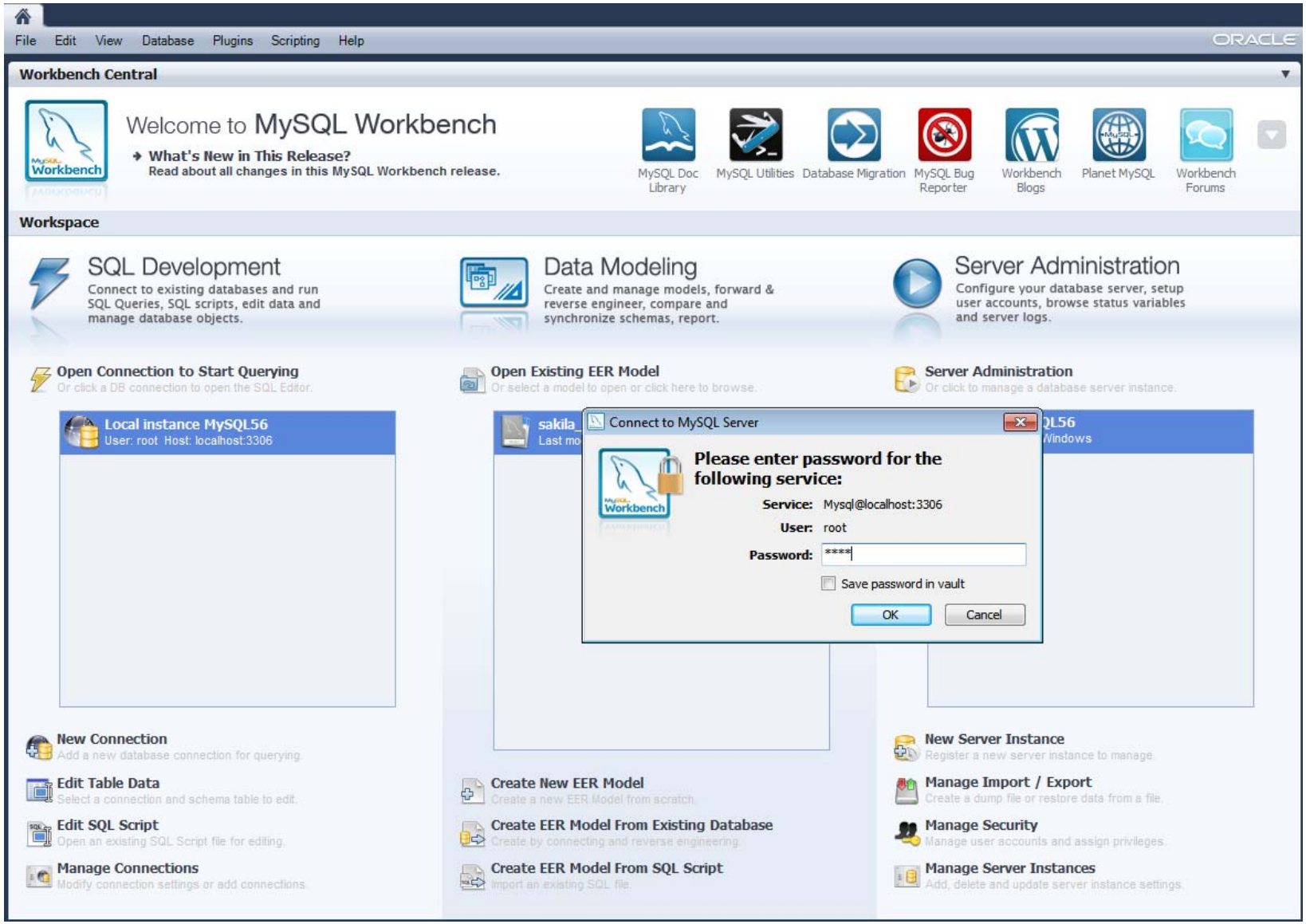



http://dev.mysql.com/doc/refman/5.6/en/mysql-installer-gui.html

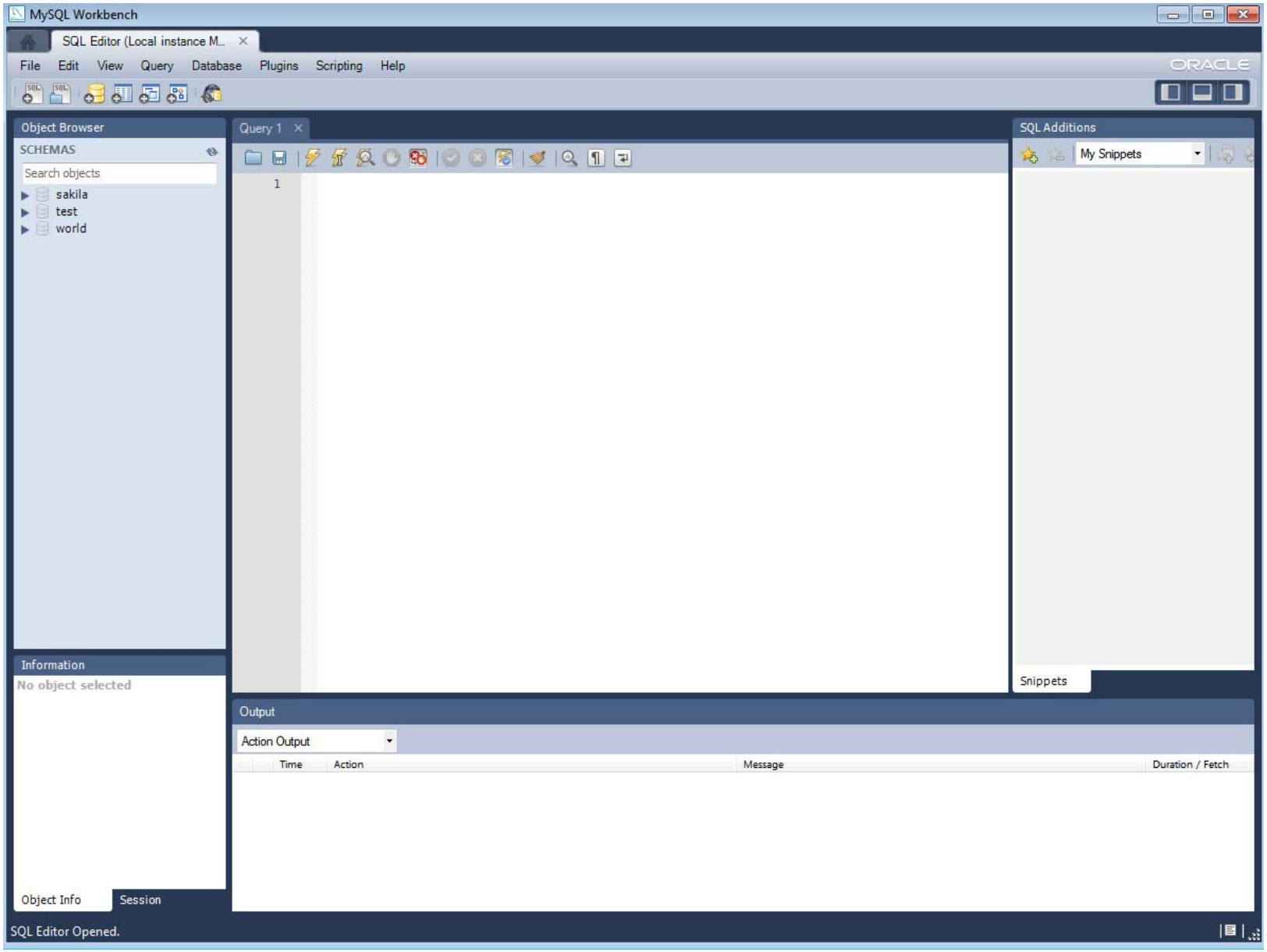
# Connecting to MySQL



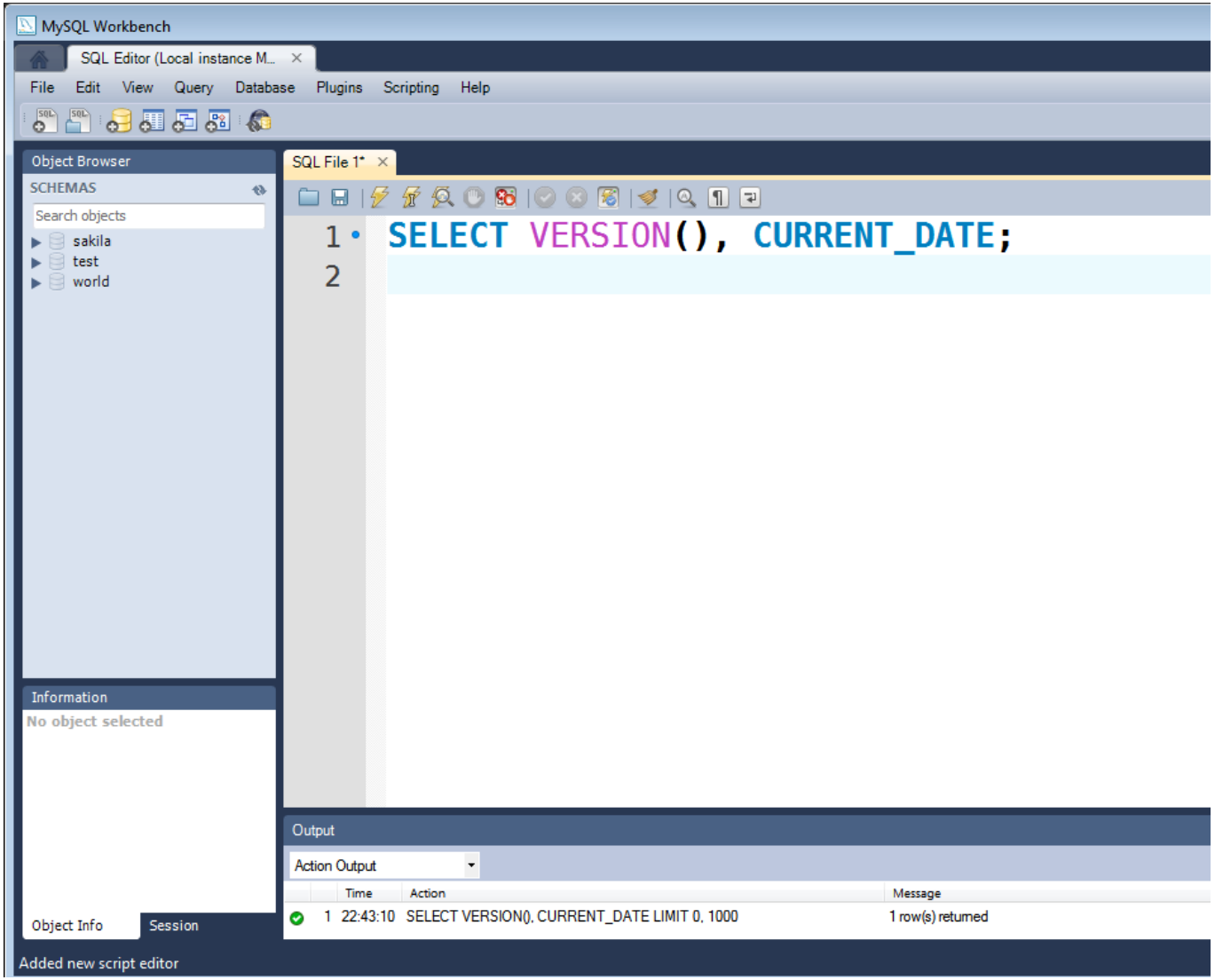
# Connecting to MySQL



# Connecting to MySQL

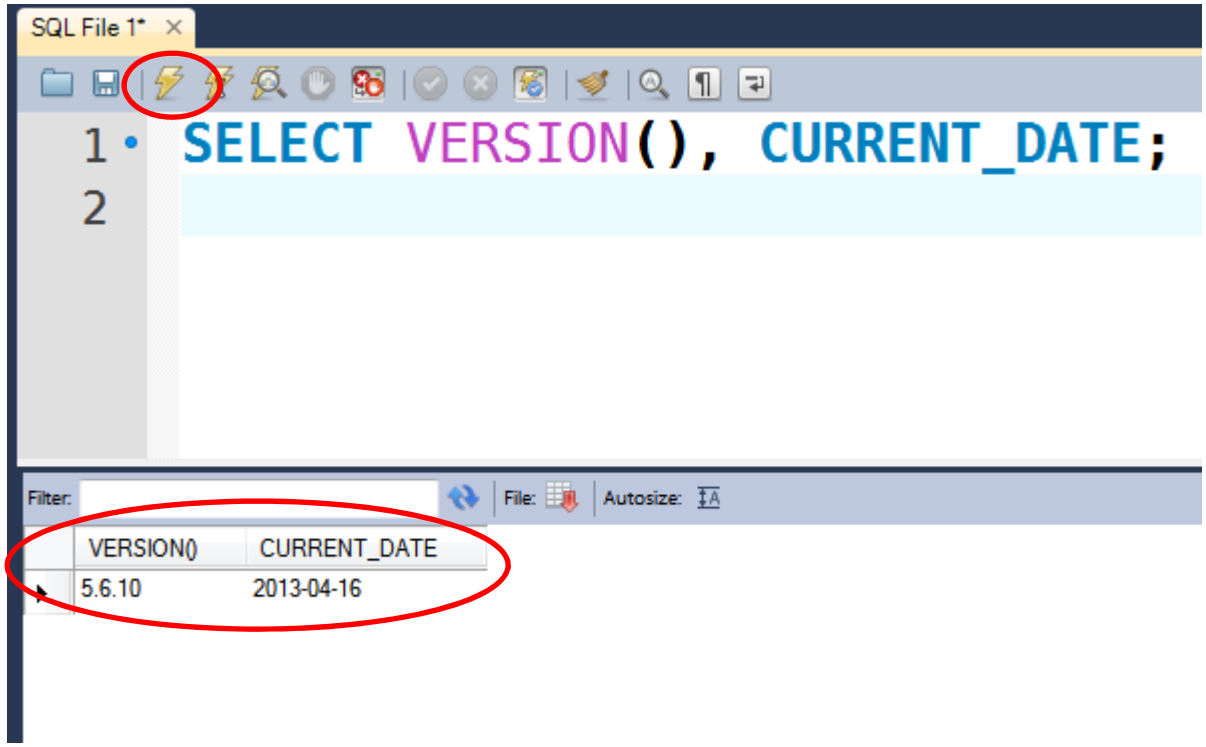


# Basic Queries




- Note that most MySQL commands end with a semicolon (;)
- MySQL returns the total number of rows found, and the total time to execute the query.

# Basic Queries



# Basic Queries

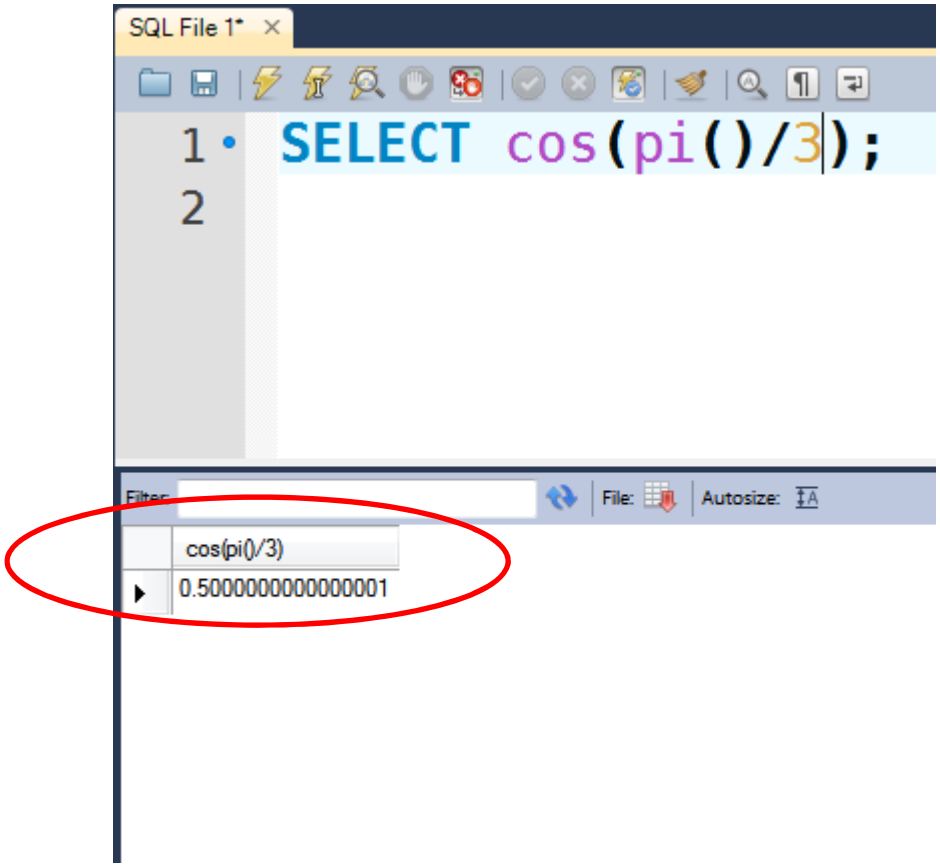
- Keywords may be entered in any letter-case.
- The following queries are equivalent:

  - ✓SELECT VERSION(), CURRENT_DATE;

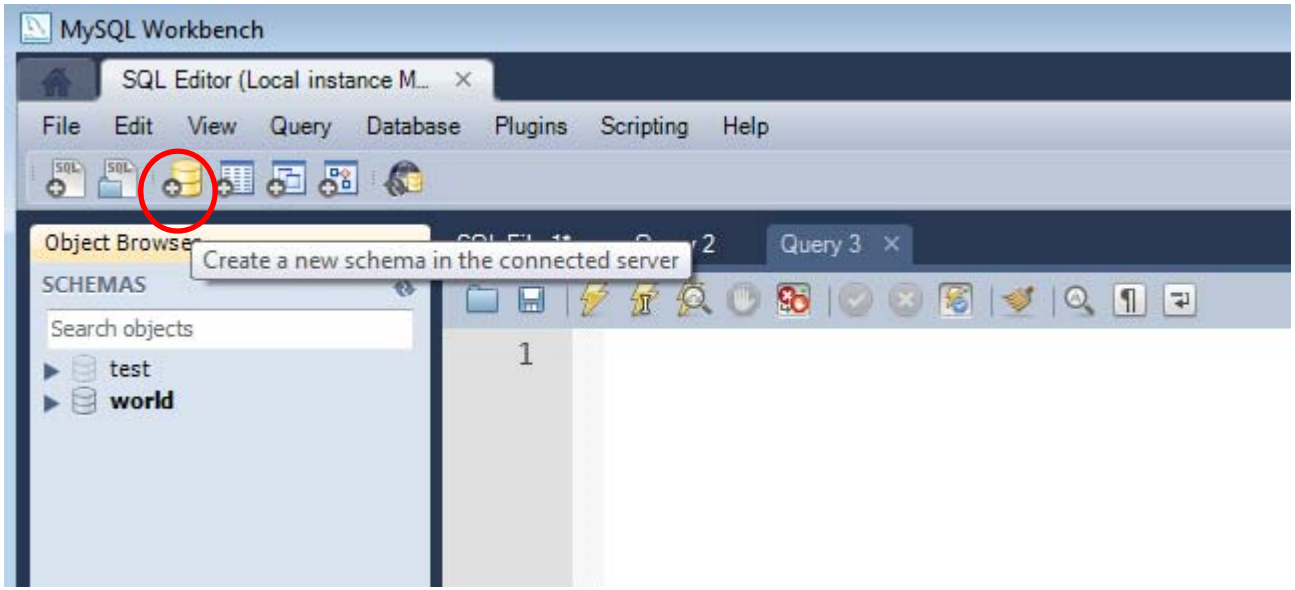
  - ✓select version(), current_date;

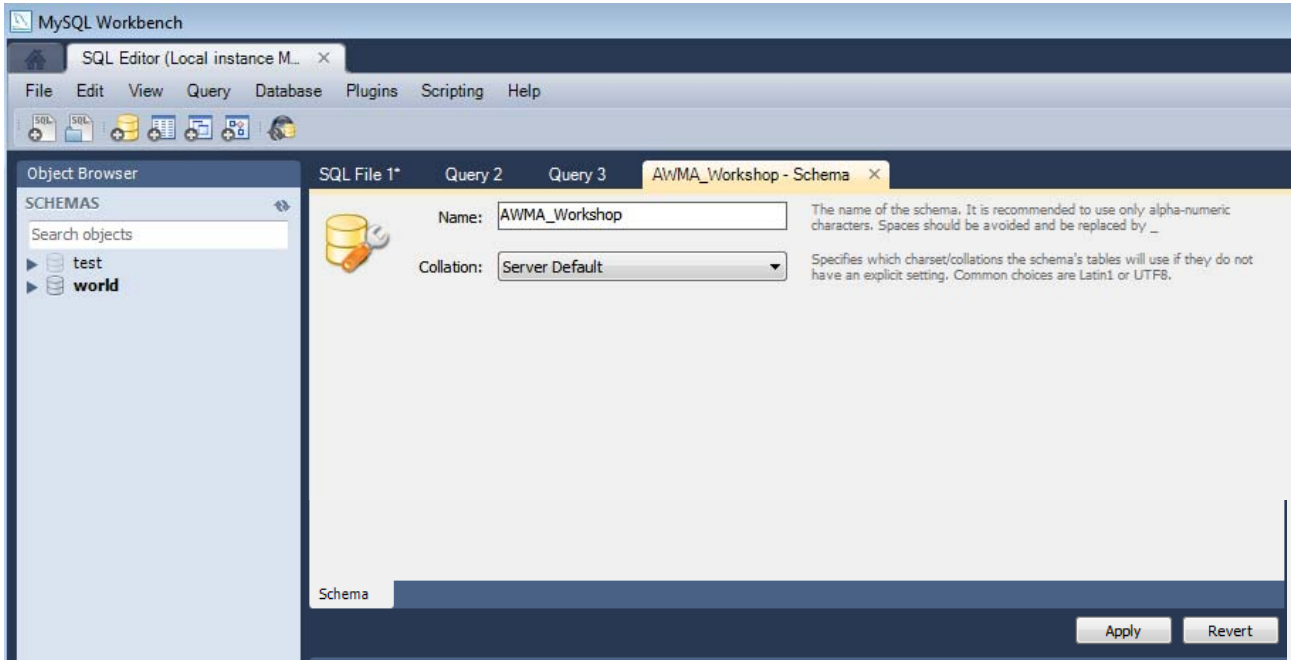
  - ✓SeLeCt vErSiOn(), current_DATE;

# Basic Queries

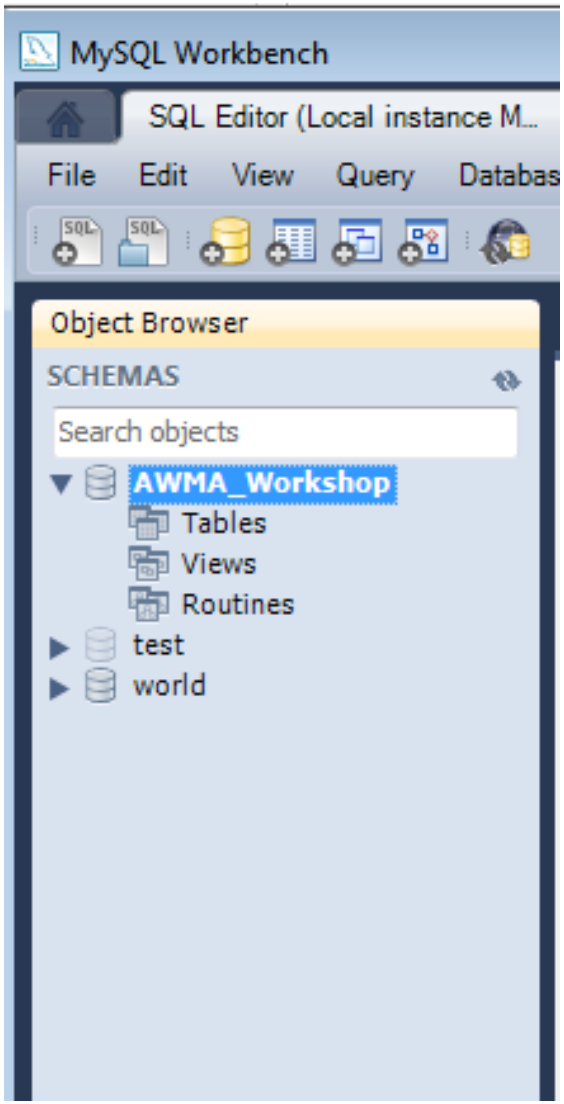


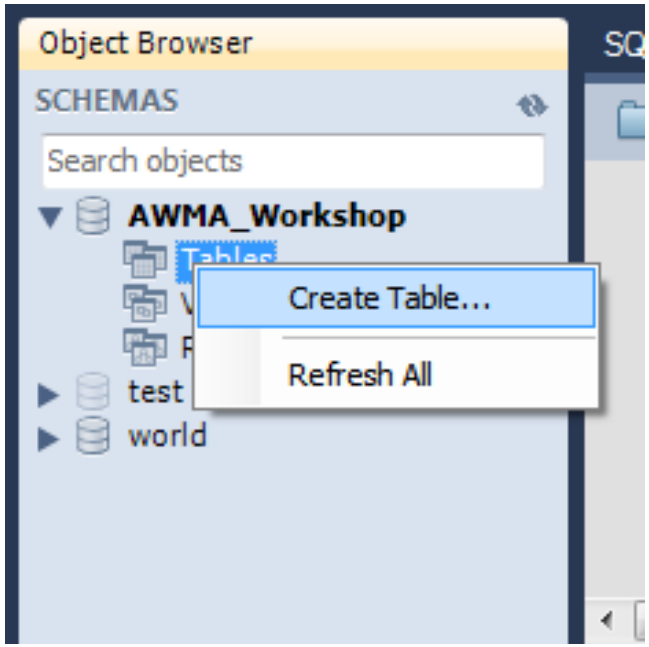
# Creating a Database



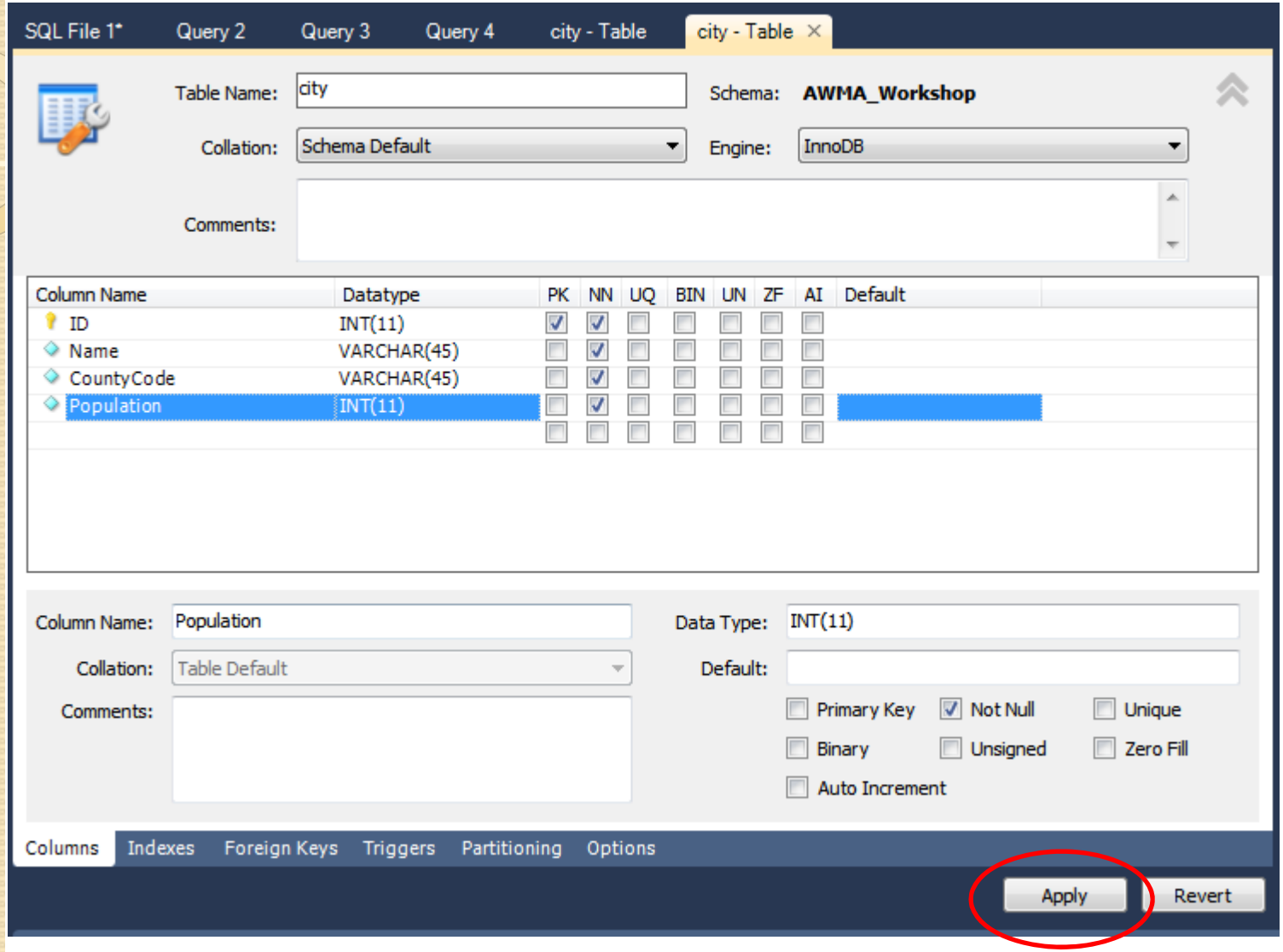



# Creating a Database

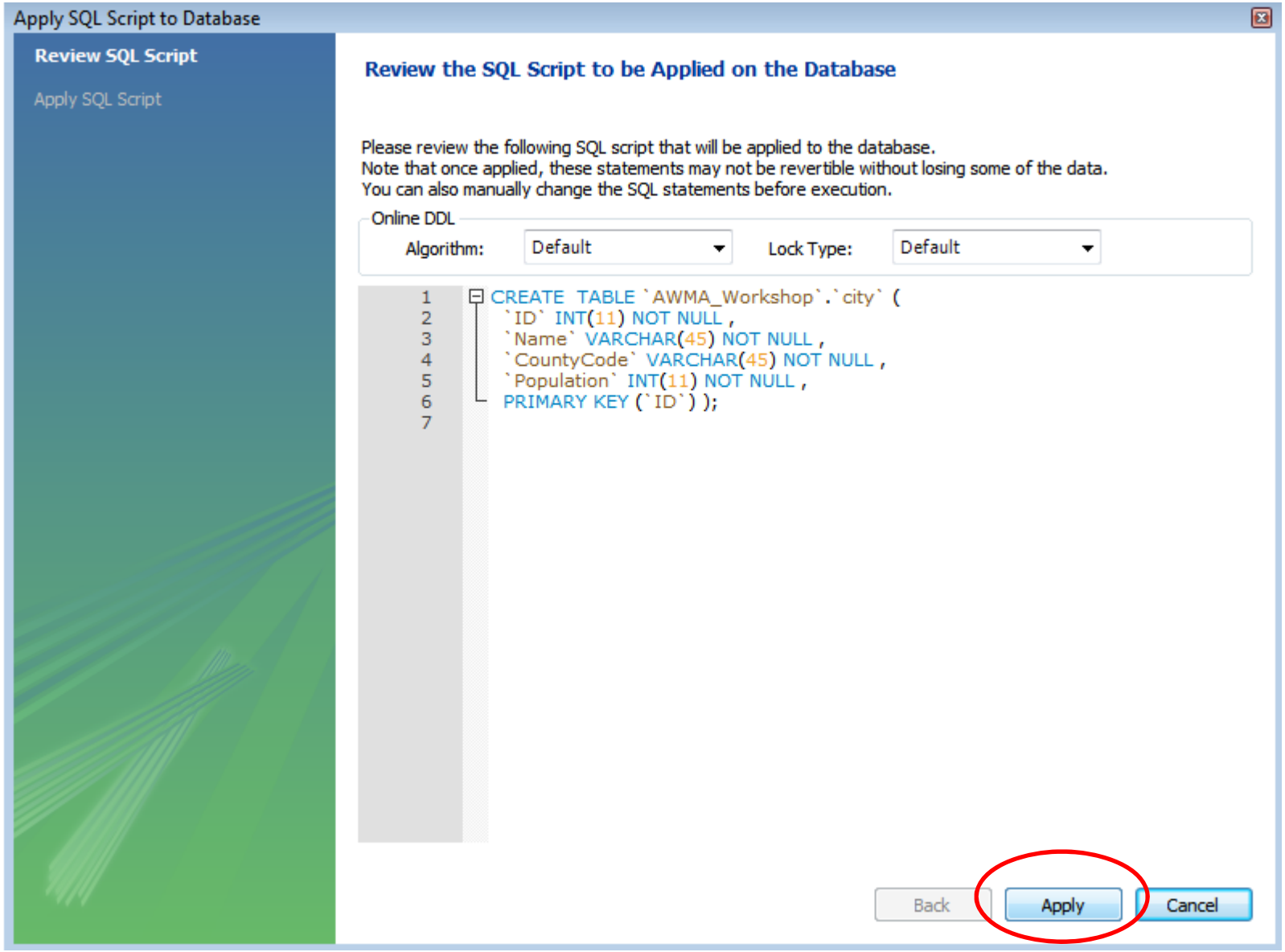


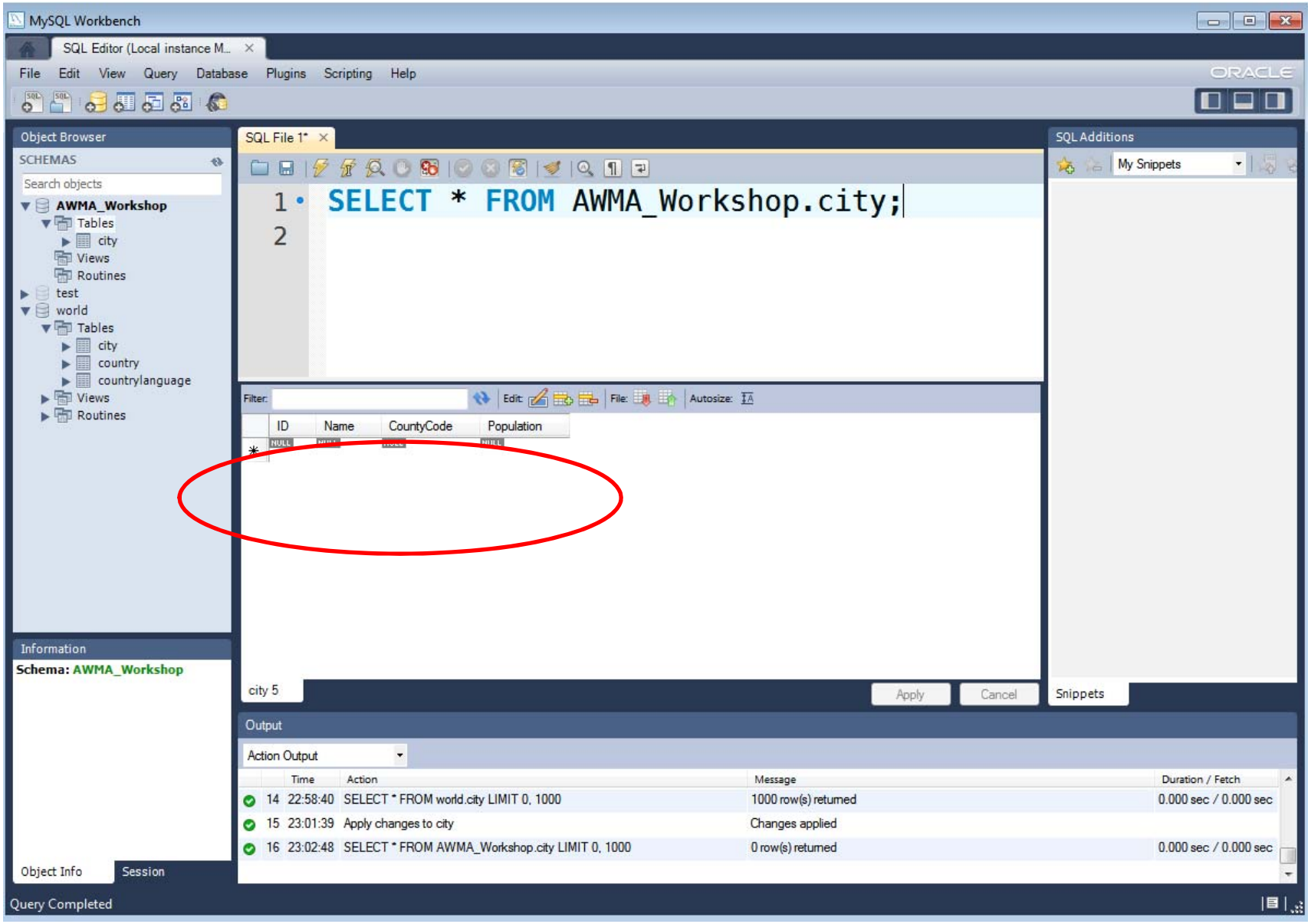
# Creating a Table



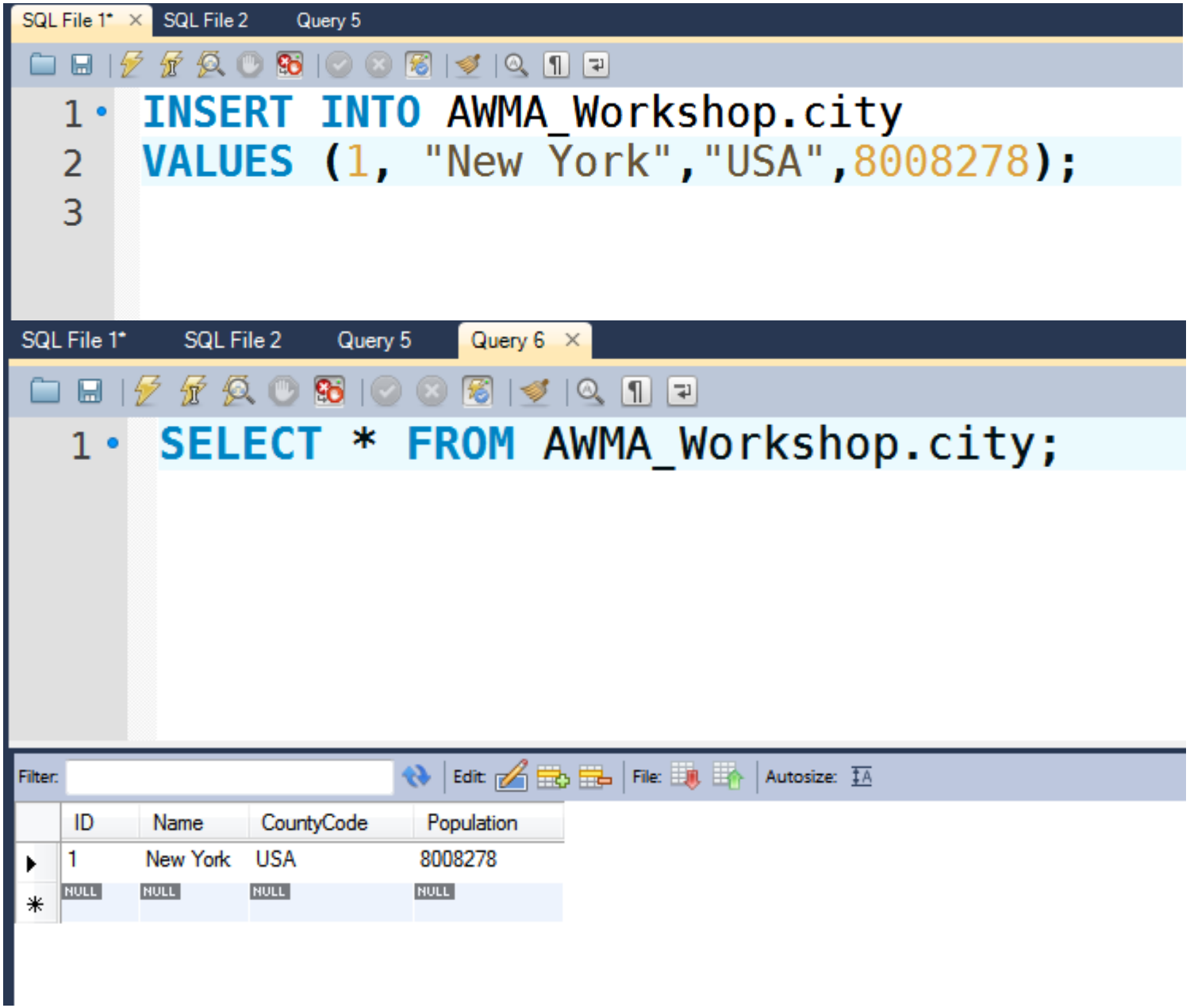
# Creating a Table



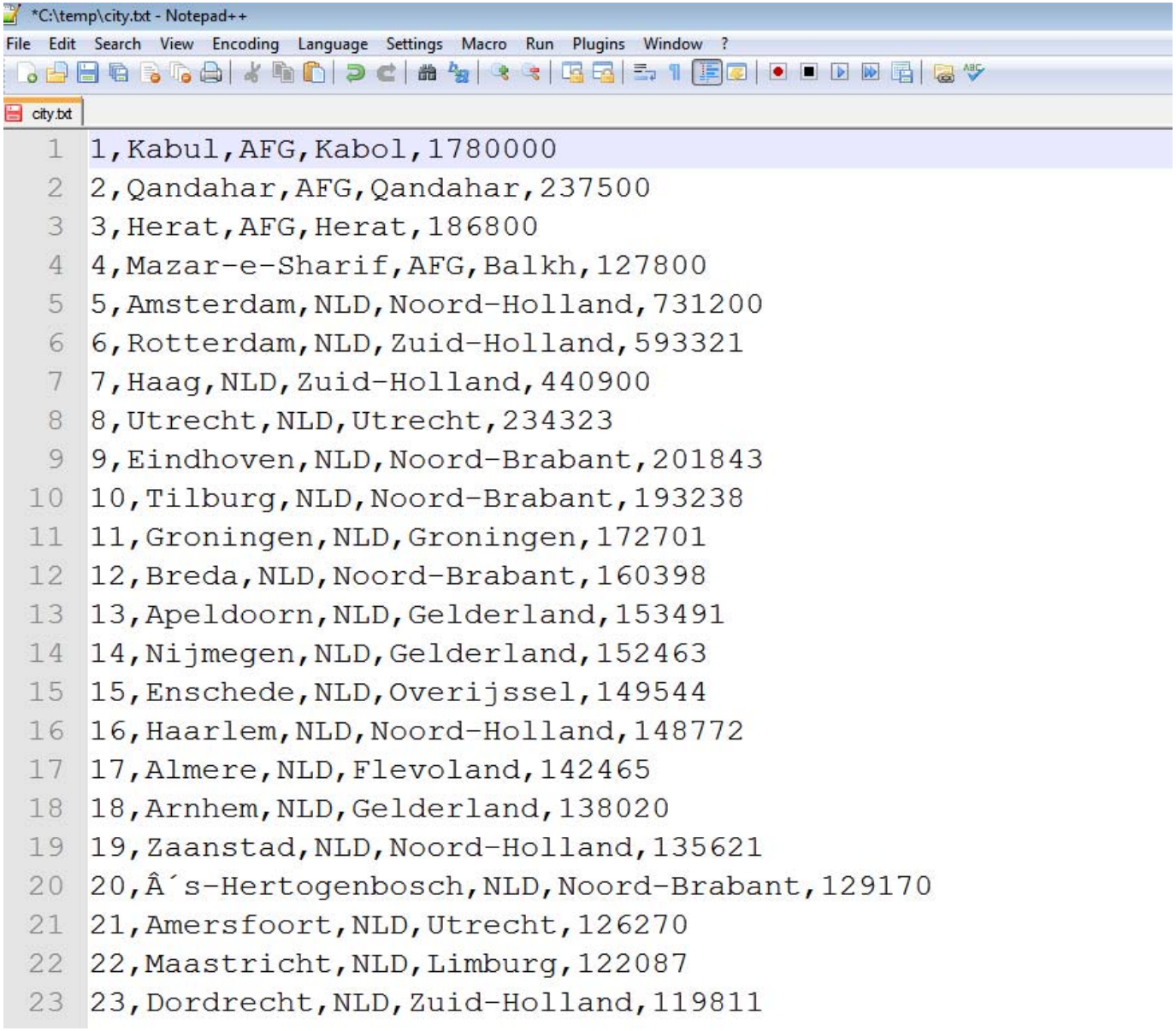
# Basic Queries



# Basic Queries



# Loading Data from a TextFile



# Loading Data from a TextFile

SQL File 1 | SQL File 2 | Query 1 | Query 2 | Query 3 | SQL File 3* | Query 4

```
1 • LOAD DATA LOCAL INFILE 'c:/temp/city.txt'
2   INTO TABLE city
3   FIELDS TERMINATED BY ","
4   LINES TERMINATED BY "\n";
```

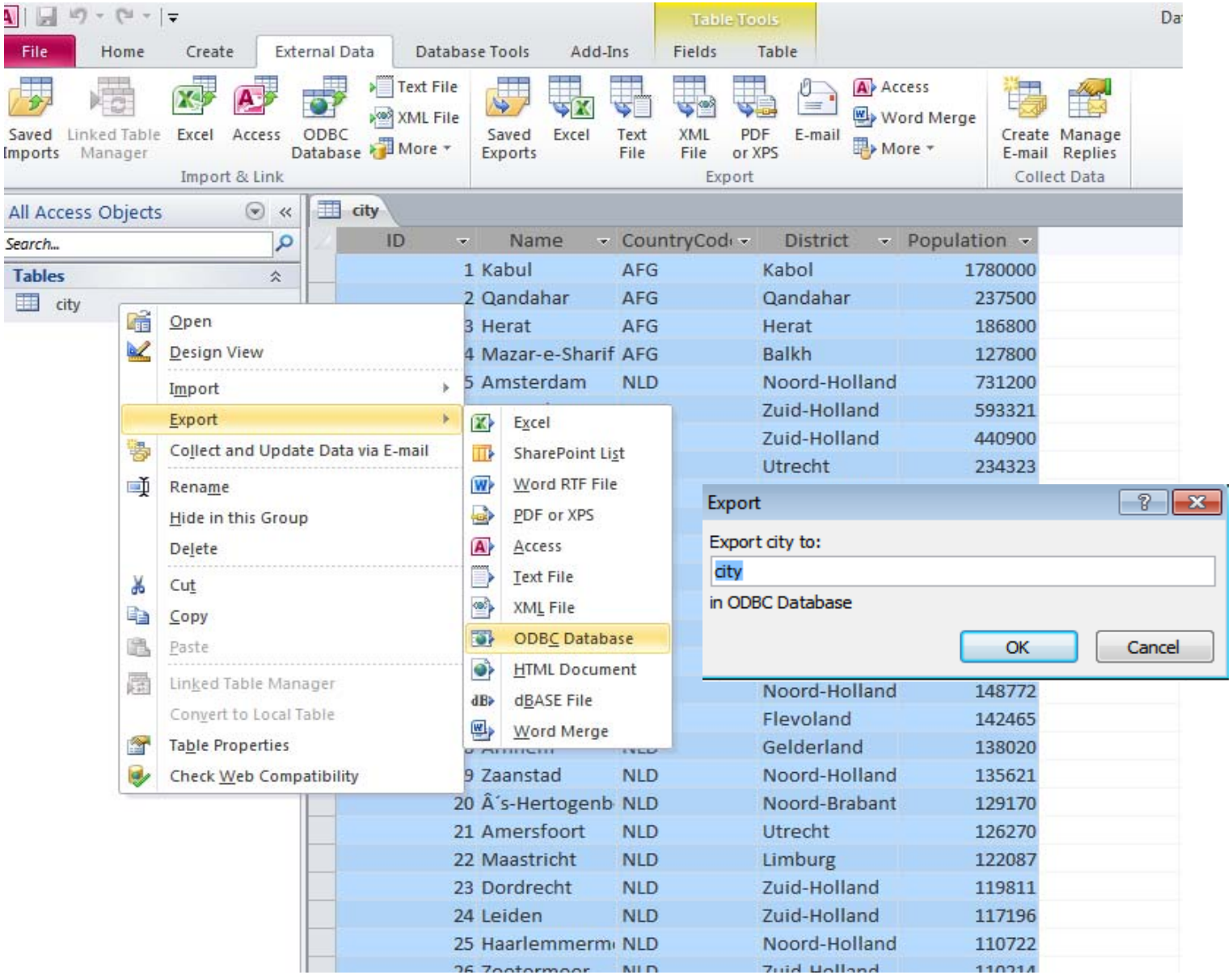
SQL File 1 | SQL File 2 | Query 1 | Query 2 | Query 3 | SQL File 3* | Query 4

```
1 • SELECT * FROM awma_workshop.city;
```

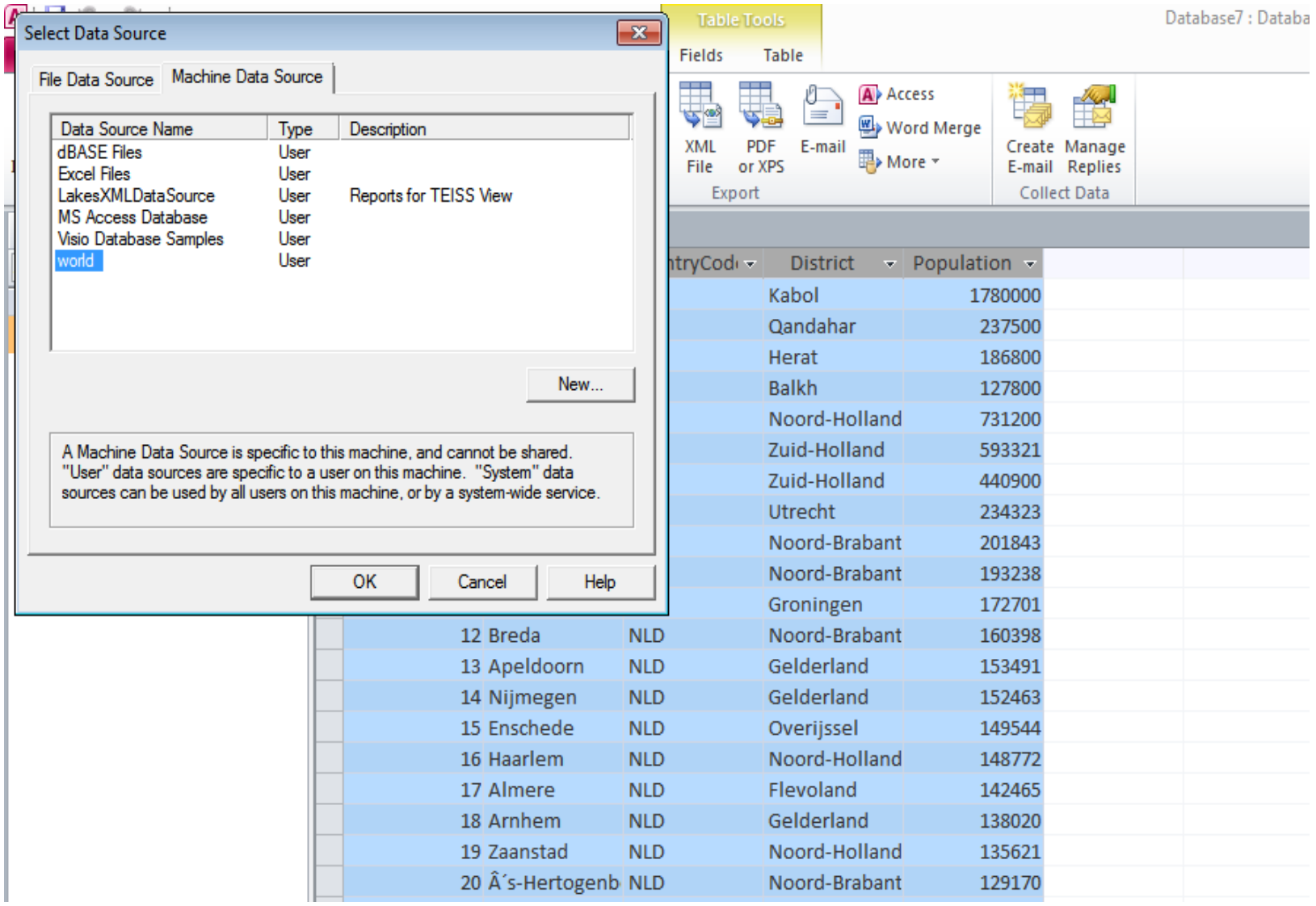
Filter: | Edit: | File: | Autosize: | Fetch rows:

| ID | Name | CountyCode | Population |
|---|---|---|---|
| 1 | Kabul | AFG | 1780000 |
| 2 | Qandahar | AFG | 237500 |
| 3 | Herat | AFG | 186800 |
| 4 | Mazar-e-Sharif | AFG | 127800 |
| 5 | Amsterdam | NLD | 731200 |
| 6 | Rotterdam | NLD | 593321 |
| 7 | Haag | NLD | 440900 |
| 8 | Utrecht | NLD | 234323 |
| 9 | Eindhoven | NLD | 201843 |
| 10 | Tilburg | NLD | 193238 |

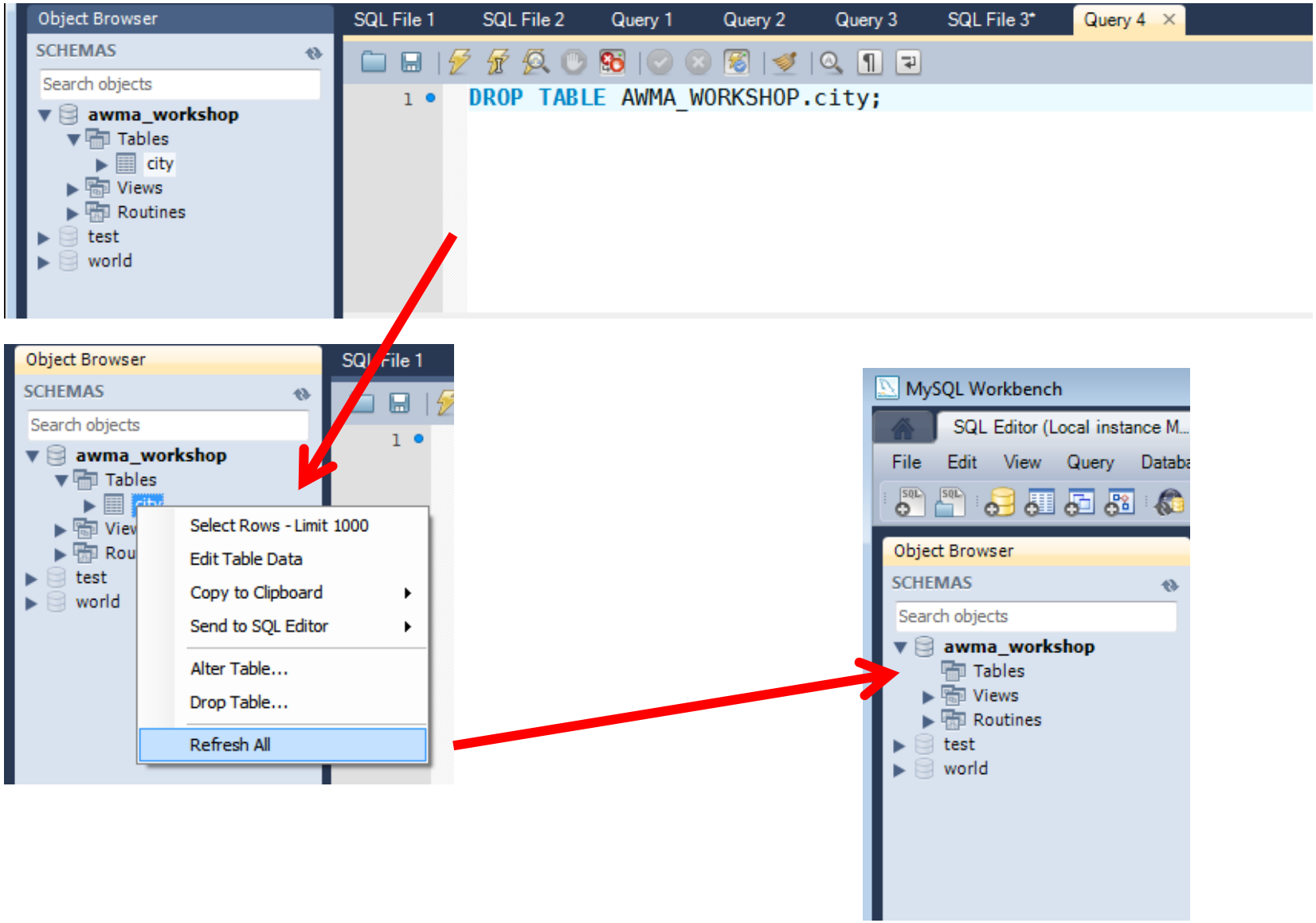
# Sending data through MS Access



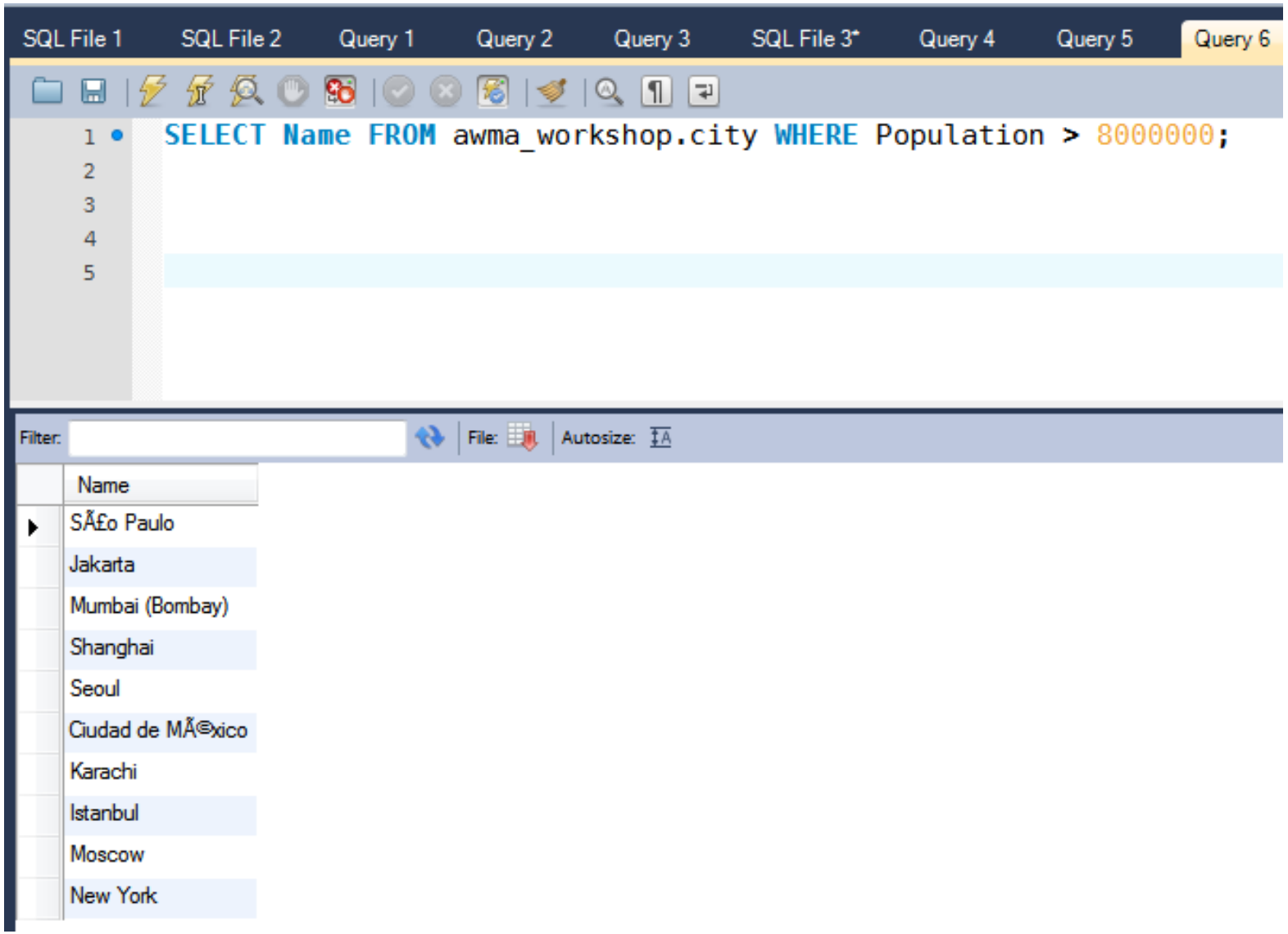
# Sending data through MS Access



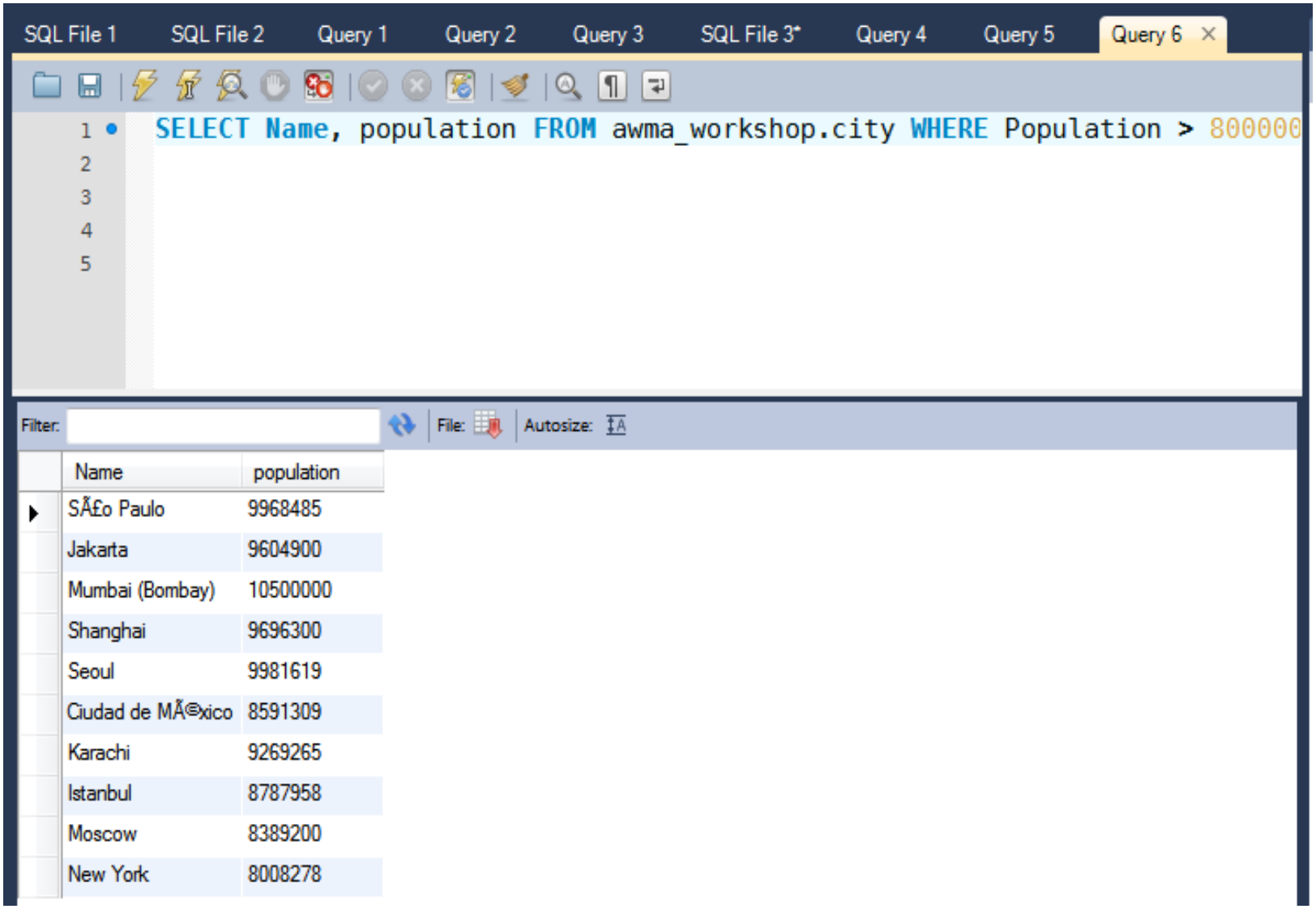
# Dropping a Table



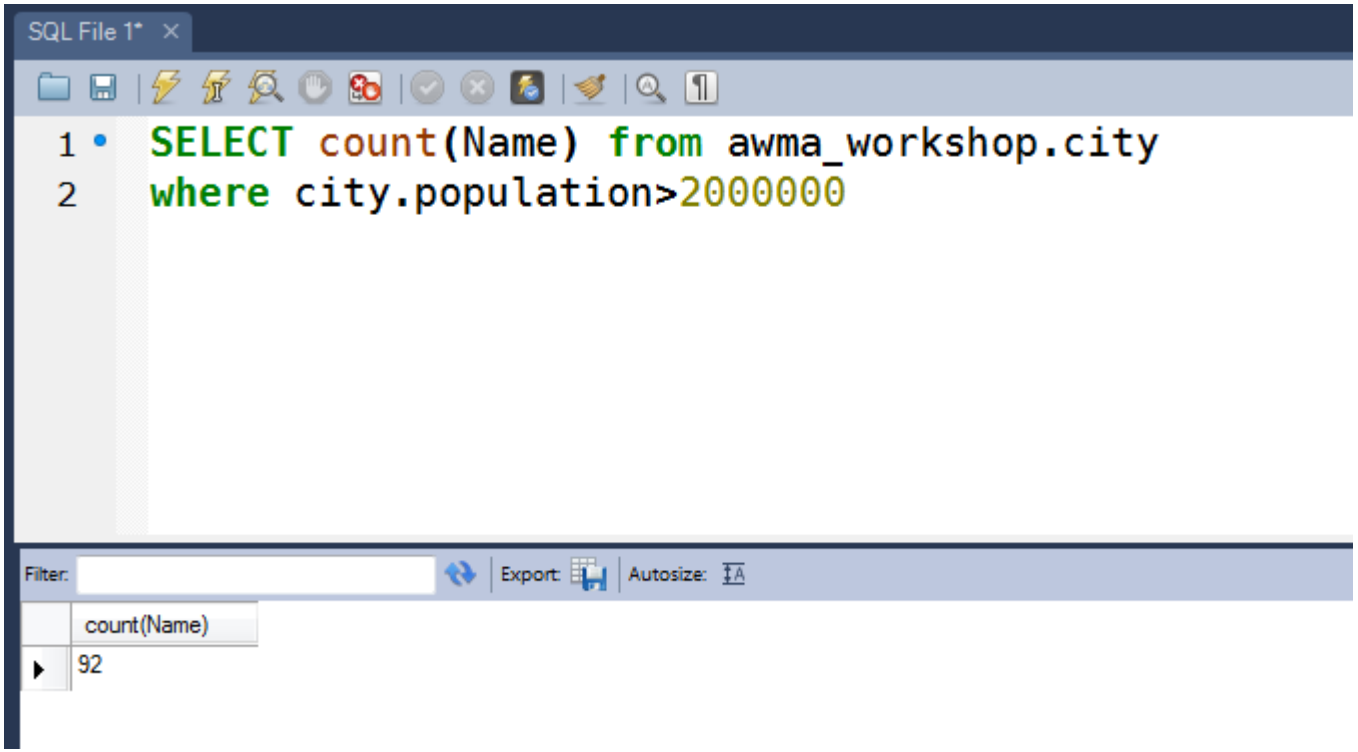
# WHERE Clause



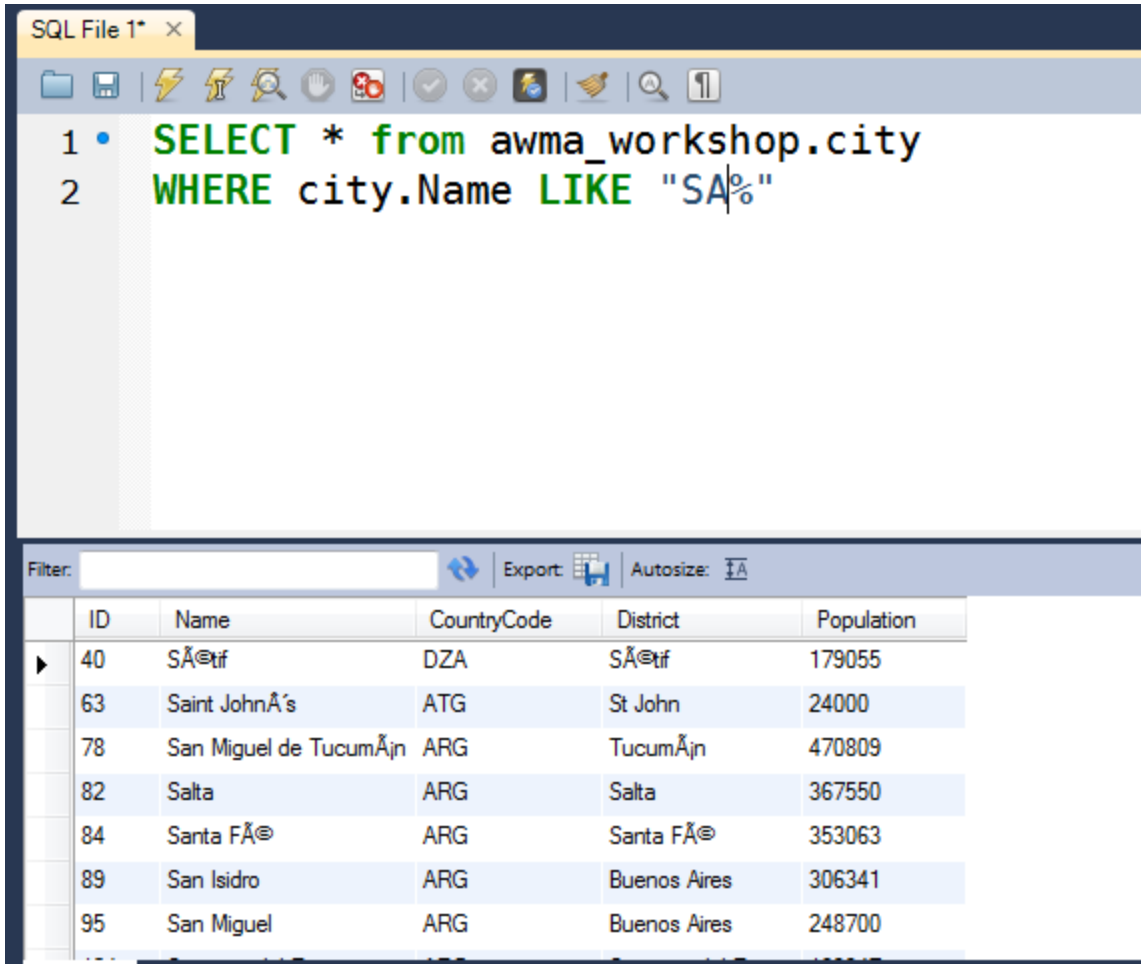
# WHERE Clause



# Counting Rows Example

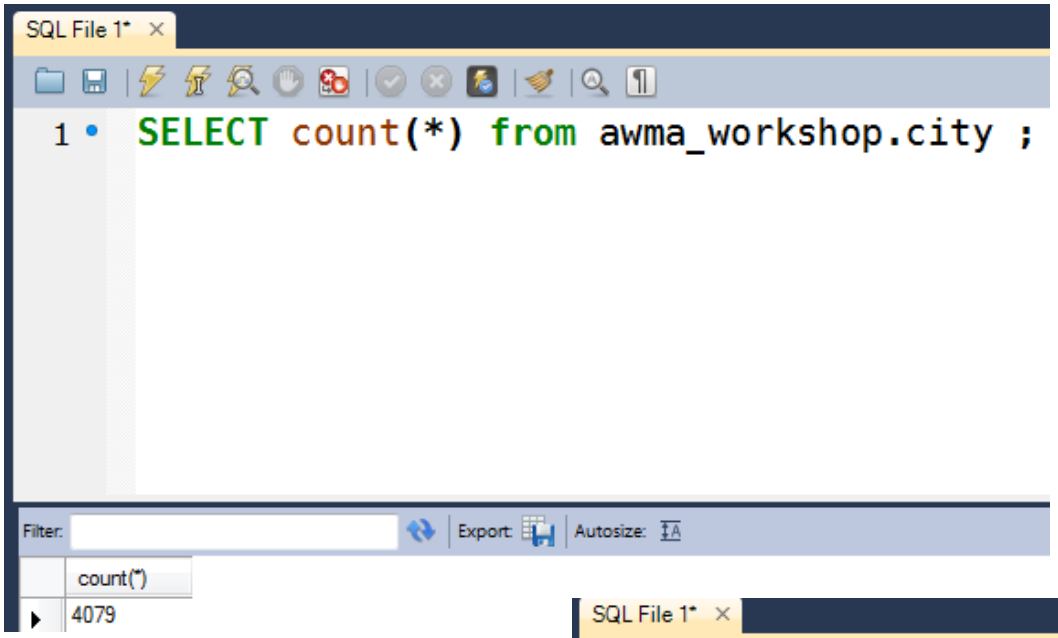


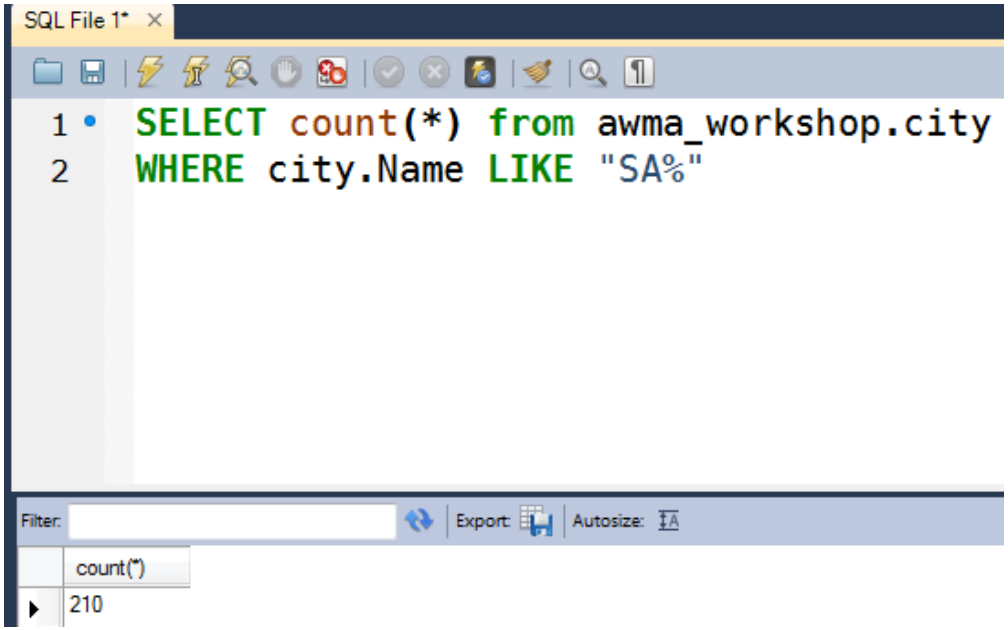
# “LIKE”




| ID | Name | CountryCode | District | Population |
|---|---|---|---|---|
| 40 | SÃ©tif | DZA | SÃ©tif | 179055 |
| 63 | Saint JohnÂ´s | ATG | St John | 24000 |
| 78 | San Miguel de TucumÃjn | ARG | TucumÃjn | 470809 |
| 82 | Salta | ARG | Salta | 367550 |
| 84 | Santa FÃ© | ARG | Santa FÃ© | 353063 |
| 89 | San Isidro | ARG | Buenos Aires | 306341 |
| 95 | San Miguel | ARG | Buenos Aires | 248700 |

# Counting Rows Example



# Counting number of cities with population larger than 20,000

```
1 • SELECT count(*) FROM awma_workshop.city;
2
3
4
```

Filter: File: Autosize:

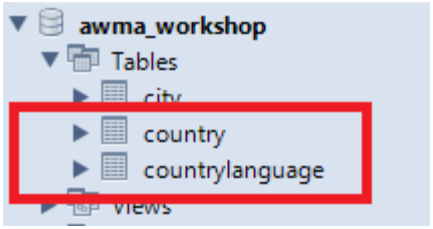
| count(*) |
| --- |
| 4079 |

```
1 • SELECT count(*) FROM awma_workshop.city WHERE Population > 20000;
2
3
4
```

Filter: File: Autosize:

| count(*) |
| --- |
| 4025 |

# Country Details



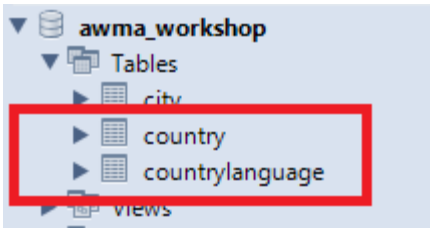

SQL File 1* | Query 5 | Query 6 | Query 7

```
1 • SELECT * FROM awma_workshop.country;
```

Filter: | Export: | Autosize:

| Code | Name | Continent | Region | SurfaceArea | IndepYear | Population | LifeExpectancy |
|---|---|---|---|---|---|---|---|
| ABW | Aruba | North America | Caribbean | 193 | NULL | 103000 | 78.4 |
| AFG | Afghanistan | Asia | Southern and Central Asia | 652090 | 1919 | 22720000 | 45.9 |
| AGO | Angola | Africa | Central Africa | 1246700 | 1975 | 12878000 | 38.3 |
| AIA | Anguilla | North America | Caribbean | 96 | NULL | 8000 | 76.1 |
| ALB | Albania | Europe | Southern Europe | 28748 | 1912 | 3401200 | 71.6 |
| AND | Andorra | Europe | Southern Europe | 468 | 1278 | 78000 | 83.5 |
| ANT | Netherlands Antilles | North America | Caribbean | 800 | NULL | 217000 | 74.7 |
| ARE | United Arab Emirates | Asia | Middle East | 83600 | 1971 | 2441000 | 74.1 |
| ARG | Argentina | South America | South America | 2780400 | 1816 | 37032000 | 75.1 |
| ARM | Armenia | Asia | Middle East | 29800 | 1991 | 3520000 | 66.4 |
| ASM | American Samoa | Oceania | Polynesia | 199 | NULL | 68000 | 75.1 |
| ATA | Antarctica | Antarctica | Antarctica | 13120000 | NULL | 0 | NULL |
| ATF | French Southern territories | Antarctica | Antarctica | 7780 | NULL | 0 | NULL |
| ATG | Antigua and Barbuda | North America | Caribbean | 442 | 1981 | 68000 | 70.5 |

# Country Language




SQL File 1* | Query 5 | Query 6 | Query 7

```
1 • SELECT * FROM awma_workshop.countrylanguage;
```

Filter: | Export: | Autosize:

| CountryCode | Language | IsOfficial | Percentage |
|---|---|---|---|
| ABW | Dutch | T | 5.3 |
| ABW | English | F | 9.5 |
| ABW | Papiamento | F | 76.7 |
| ABW | Spanish | F | 7.4 |
| AFG | Balochi | F | 0.9 |
| AFG | Dari | T | 32.1 |
| AFG | Pashto | T | 52.4 |
| AFG | Turkmenian | F | 1.9 |
| AFG | Uzbek | F | 8.8 |
| AGO | Ambo | F | 2.4 |
| AGO | Chokwe | F | 4.2 |
| AGO | Kongo | F | 13.2 |
| AGO | Luchazi | F | 2.4 |
| AGO | Luimbe-ngan... | F | 5.4 |

# Selecting Distinct *Language*

```
1 • SELECT DISTINCT language FROM awma_workshop.countrylanguage;
2
3
```

Filter: | File: | Autosize:

| language |
|---|
| Dutch |
| English |
| Papiamento |
| Spanish |
| Balochi |
| Dari |
| Pashto |
| Turkmenian |
| Uzbek |
| Ambo |
| Chokwe |
| Kongo |
| Luchazi |
| Luimbe-nganguela |
| Luvale |
| Mbundu |
| Nyaneka-nkhumbi |
| Ovimbundu |
| Albaniana |
| Greek |
| Macedonian |
| Catalan |
| French |

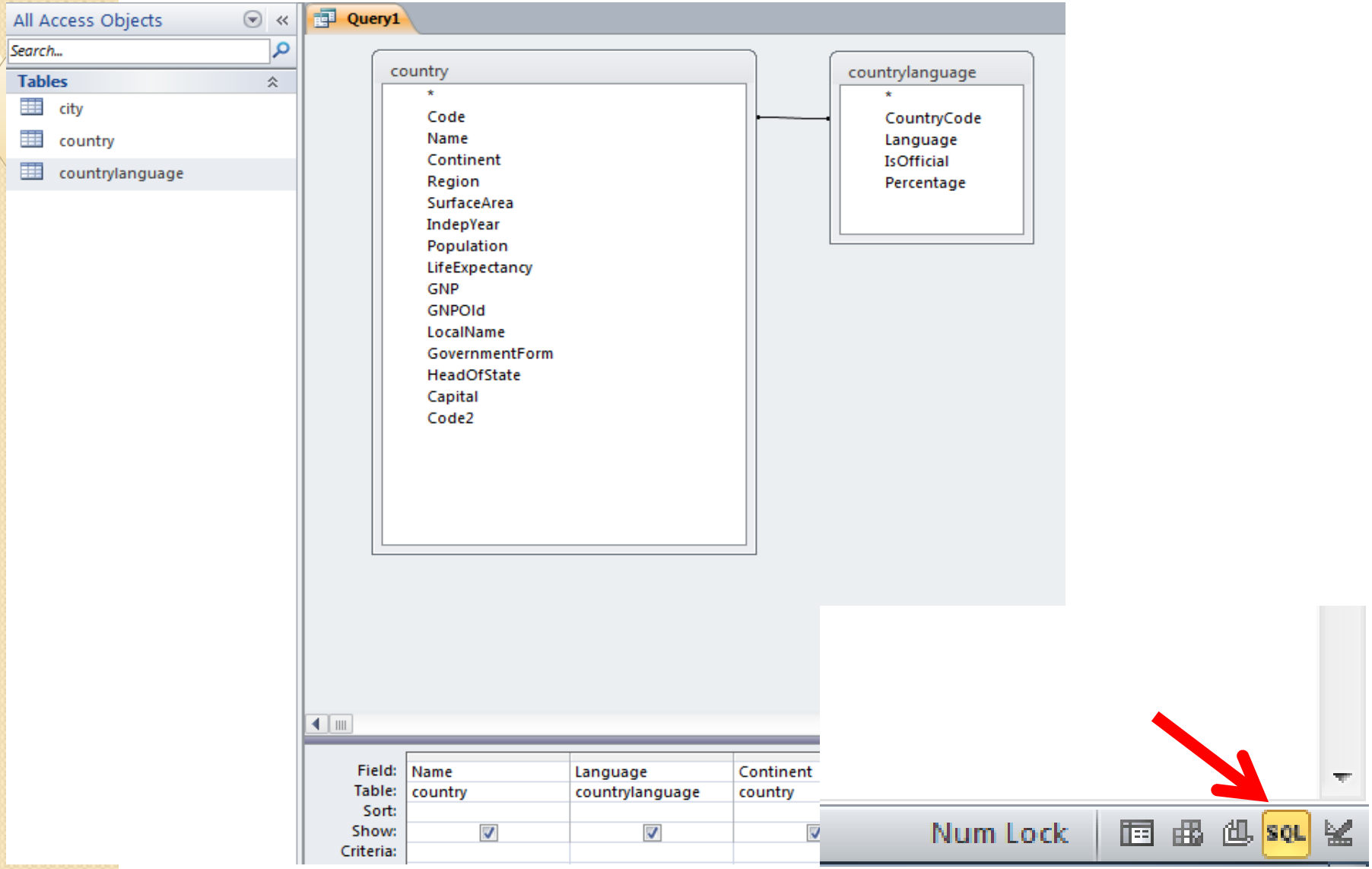
countrylanguage 7

Output

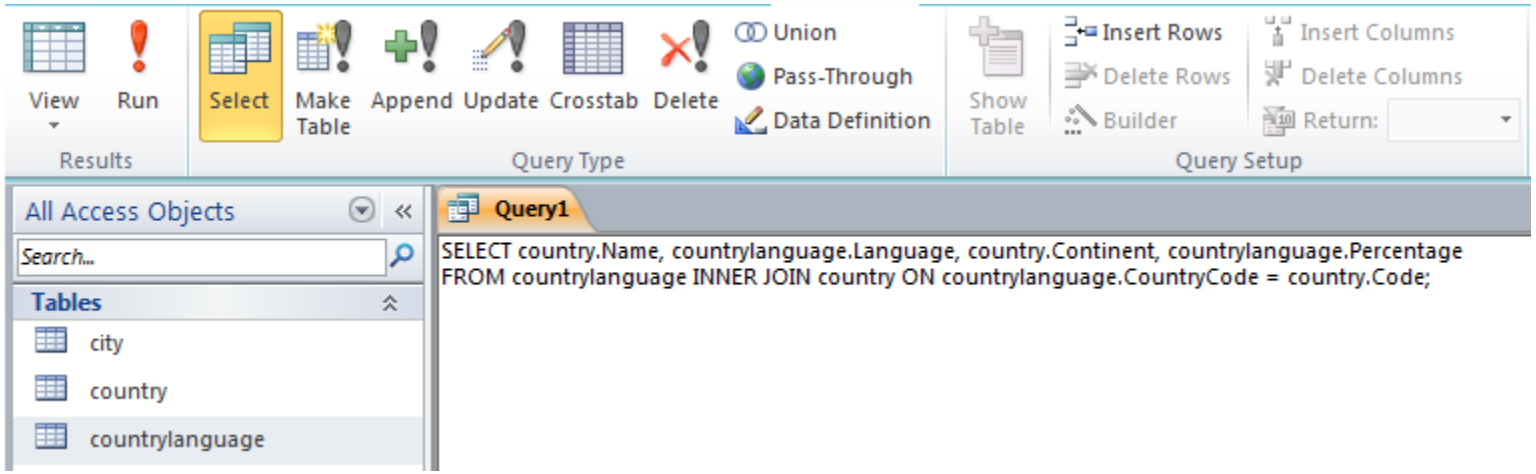
Action Output

| | Time | Action | Message |
|---|---|---|---|
| 1 | 00:00:49 | SELECT DISTINCT language FROM awma_workshop.countrylanguage LIMIT 0, 1000 | 457 row(s) returned |

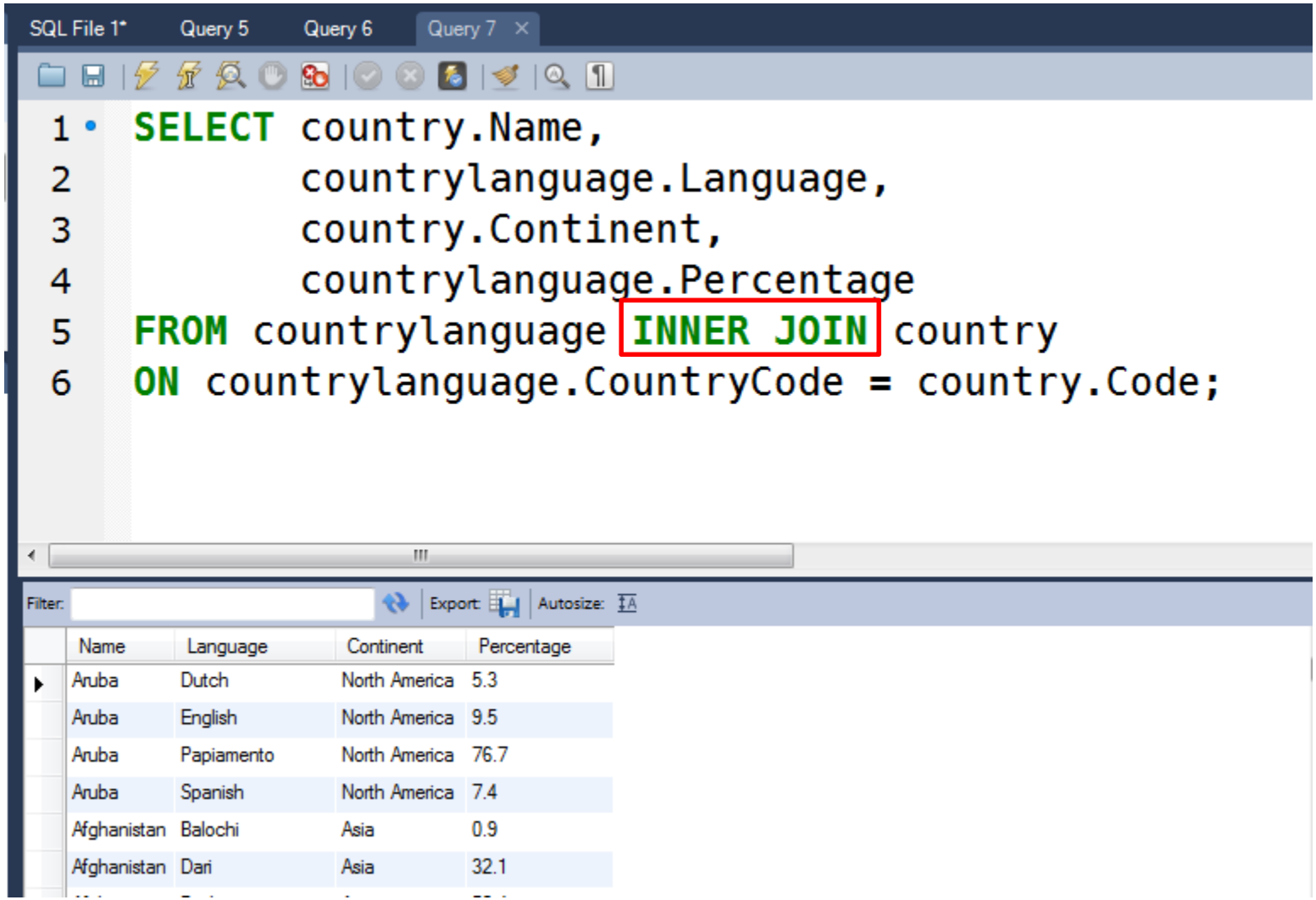
# Joining two table



# Joining two table



# Joining two table



# Joining two table *USING* ( )

```
1 • SELECT a.Name,b.language, b.percentage from
2   city as a inner join countrylanguage as b USING (CountryCode);
3
```

Filter: File: Autosize: Fetch rows:

| Name | language | percentage |
|---|---|---|
| Kabul | Balochi | 0.9 |
| Kabul | Dari | 32.1 |
| Kabul | Pashto | 52.4 |
| Kabul | Turkmenian | 1.9 |
| Kabul | Uzbek | 8.8 |
| Qandahar | Balochi | 0.9 |
| Qandahar | Dari | 32.1 |
| Qandahar | Pashto | 52.4 |
| Qandahar | Turkmenian | 1.9 |
| Qandahar | Uzbek | 8.8 |
| Herat | Balochi | 0.9 |
| Herat | Dari | 32.1 |
| Herat | Pashto | 52.4 |
| Herat | Turkmenian | 1.9 |
| Herat | Uzbek | 8.8 |
| Mazar-e-Sharif | Balochi | 0.9 |
| Mazar-e-Sharif | Dari | 32.1 |
| Mazar-e-Sharif | Pashto | 52.4 |
| Mazar-e-Sharif | Turkmenian | 1.9 |
| Mazar-e-Sharif | Uzbek | 8.8 |
| Amsterdam | Arabic | 0.9 |
| Amsterdam | Dutch | 95.6 |
| Amsterdam | Fries | 3.7 |

Result 10

Output

Action Output

| | Time | Action | Message |
|---|---|---|---|
| 1 | 00:03:59 | SELECT a.Name,b.language, b.percentage from city as a inner join countrylanguage as b USING (CountryCode) LIMIT 0, 1000 | 1000 row(s) returned |