Databases and SQL Editor in PhpStorm

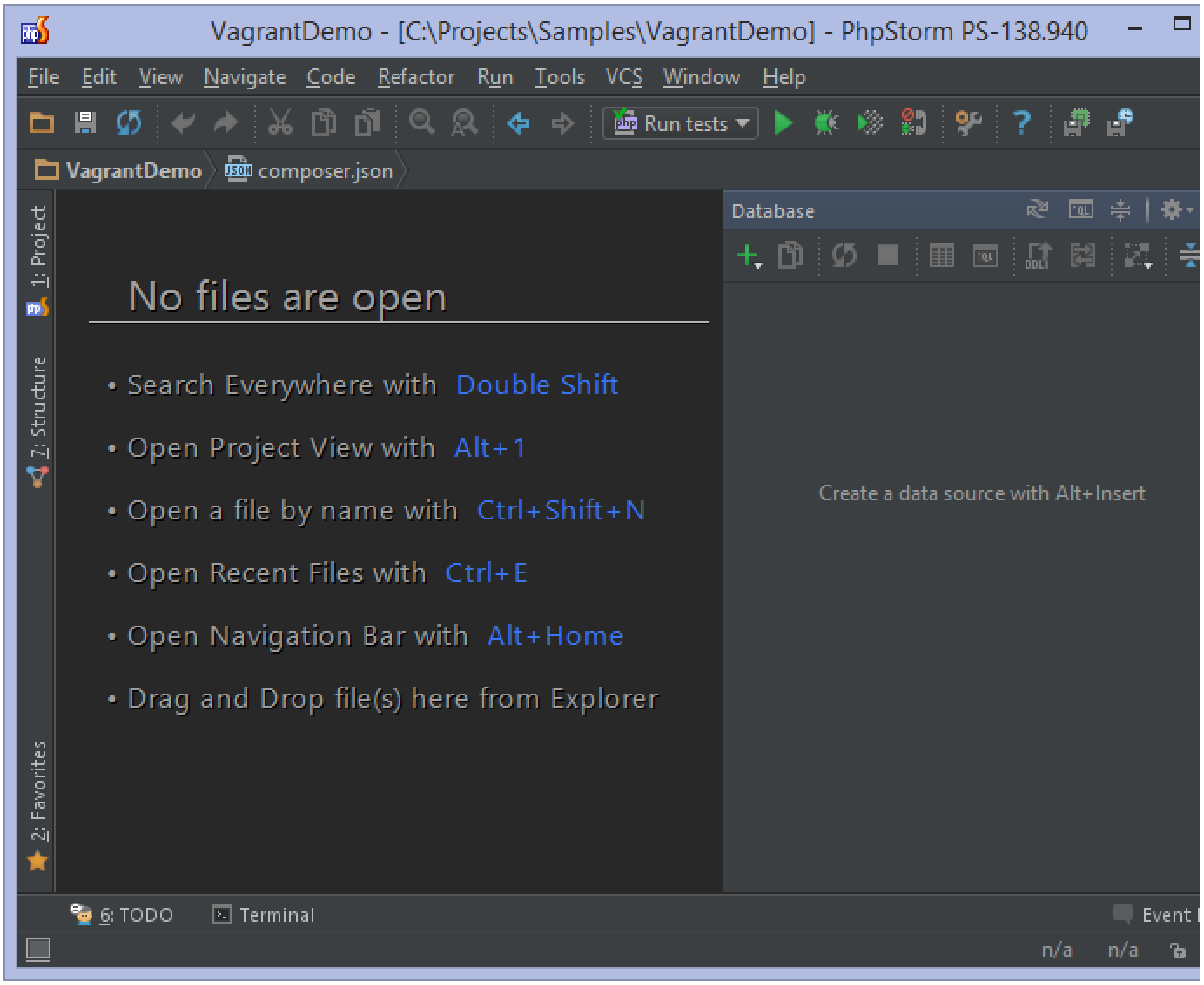
Redirection Notice
This page will redirect to https://www.jetbrains.com/help/phpstorm/relational-databases.html in about 2 seconds.

Tweet

When writing an application, chances are that we have to work with a database. PhpStorm can help us perform all types of routine database tasks, such as querying for a record; checking what that column was named again; database development where we have to create the schema structure; or just developing PHP code against a database. In this tutorial, we will see how.

Working with the Database Tool Window

Database support is provided through the Database Tool Window on the right-hand side of the IDE. Use the View | Tool Windows | Database menu or Find Action, by pressing Ctrl+Alt+A (Alt-CMD-A on Mac OS X) and search for "Database" to open it.



No files are open

- Search Everywhere with Double Shift
- Open Project View with Alt+1
- Open a file by name with Ctrl+Shift+N
- Open Recent Files with Ctrl+E
- Open Navigation Bar with Alt+Home
- Drag and Drop file(s) here from Explorer

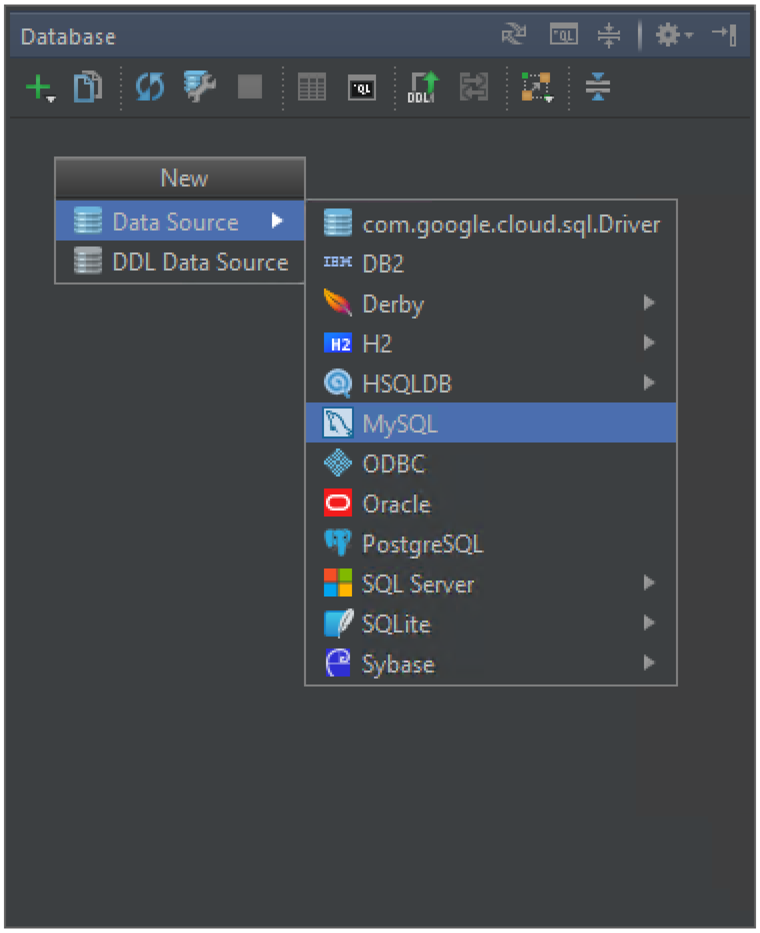
Create a data source with Alt+Insert

From the Database tool window, we can connect to various data sources and view database structure; modify it by refactoring tables and columns; manage data stored in tables by simply editing or adding values; write arbitrary SQL queries with code completion and syntax highlighting; and run SQL queries from code or the built-in SQL editor. We can generate UML diagrams showing relations between tables, copy records as INSERT statements, get smart completion when writing joins on tables, and so on.

But first things first: let's set up a database connection.

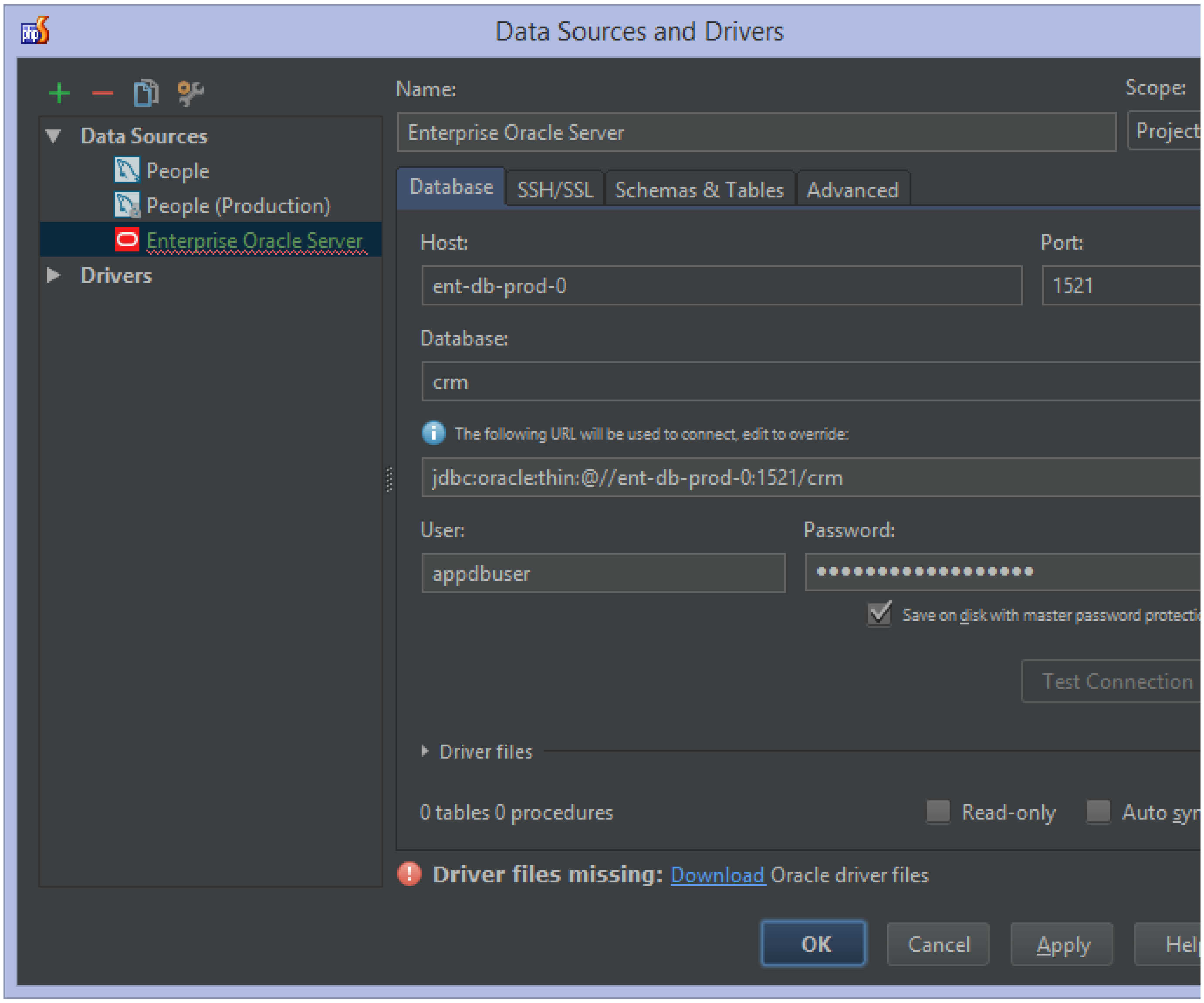
Setting up a database connection

From the Database Tool window, we can use the green + icon in the toolbar or press Alt+Insert (CMD-N on Mac OS X) to create a new data source. A data source can be a "real" data source, connected to a database system, or a "DDL" data source which is a collection of files that describe tables, indexes and such and can be exported to a database server later on.



To connect to a database, we can first pick the type of database. We can connect to Google Cloud SQL, DB2, Derby, H2, HSQLDB, MySQL, an ODBC connection, Oracle, PostgreSQL, SQL Server (and Microsoft Azure Database), or SQLite an Sybase.

For every database system, different options will be available for configuring the connection. Typically these will be the server, database name, username and password, but many drivers provide additional options for configuring the connection. PhpStorm does not ship with all database drivers installed, but it does provide a handy way of downloading them when needed: click the "Download ... driver files" link next to the warning about missing drivers to download them.


Data Sources and Drivers

Data Sources
- People
- People (Production)
- Enterprise Oracle Server

Drivers

Name: Enterprise Oracle Server

Scope: Project

Host: ent-db-prod-0

Port: 1521

Database: crm

The following URL will be used to connect, edit to override:

jdbc:oracle:thin:@//ent-db-prod-0:1521/crm

User: appdbuser

Password: ••••••••••••••••••

✓ Save on disk with master password protecti

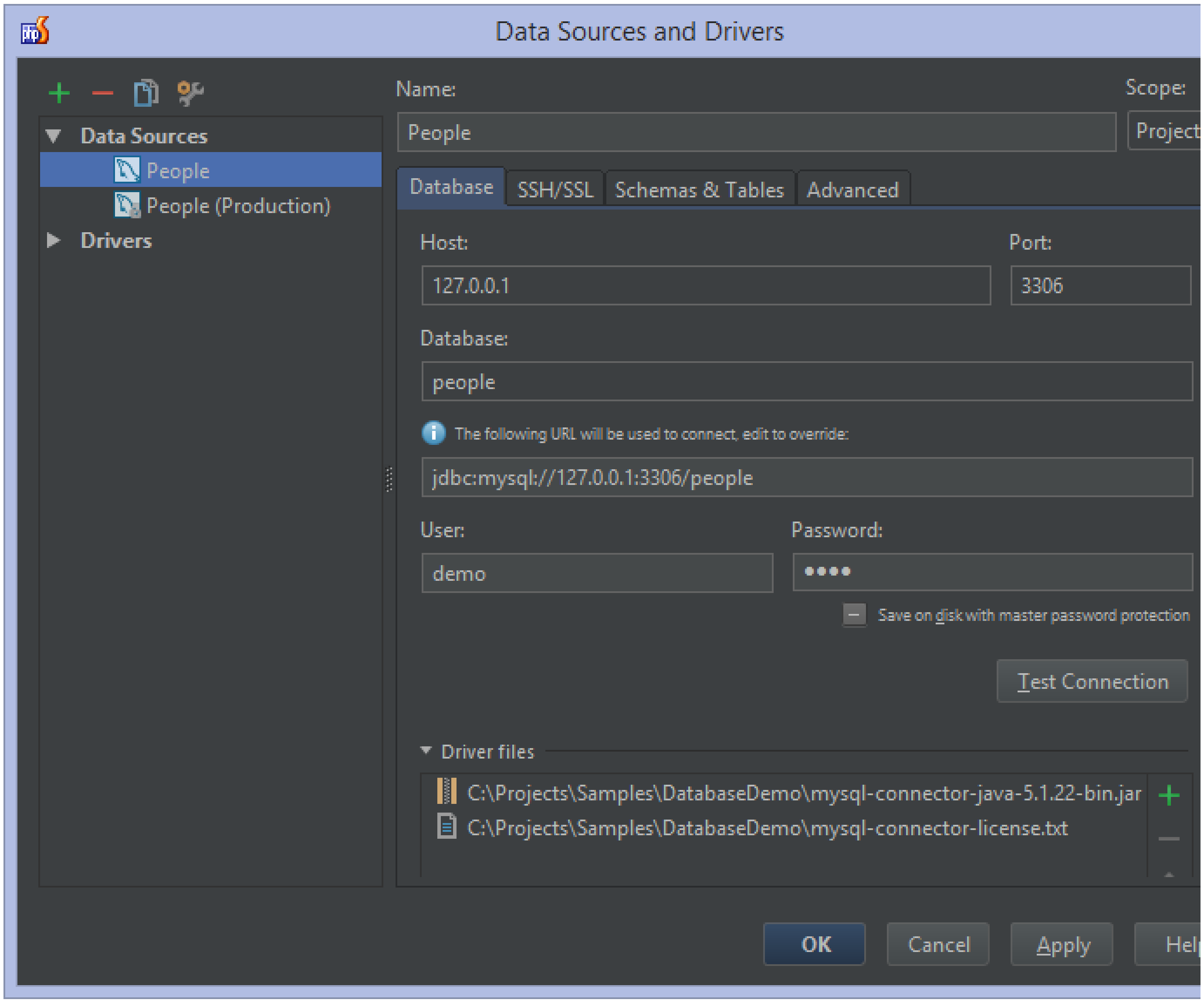
0 tables 0 procedures

Read-only

Auto syn

Driver files missing: Download Oracle driver files

Here's another example for MySQL connecting to a local development database:



Name: People

Scope: Project

Host: 127.0.0.1

Port: 3306

Database: people

The following URL will be used to connect, edit to override:

jdbc:mysql://127.0.0.1:3306/people

User: demo

Password: ••••

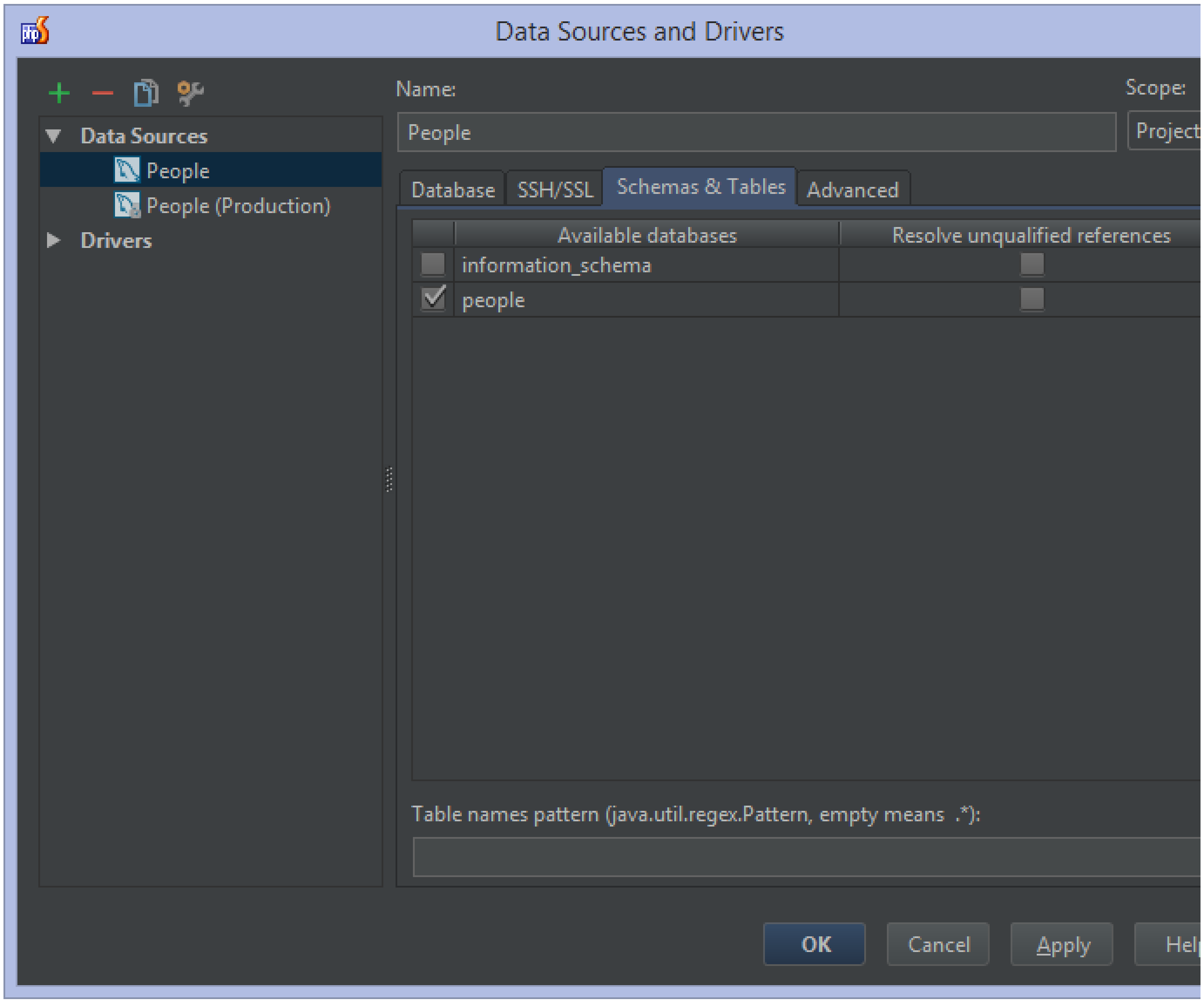
Save on disk with master password protection

C:\Projects\Samples\DatabaseDemo\mysql-connector-java-5.1.22-bin.jar

C:\Projects\Samples\DatabaseDemo\mysql-connector-license.txt

For most drivers, we can just enter the connection details, and PhpStorm will generate the underlying database connection string for us. For some, we may have to enter a full JDBC connection string. Try searching the Internet for "<driver/database type> jdbc connection string" if you encounter such driver.

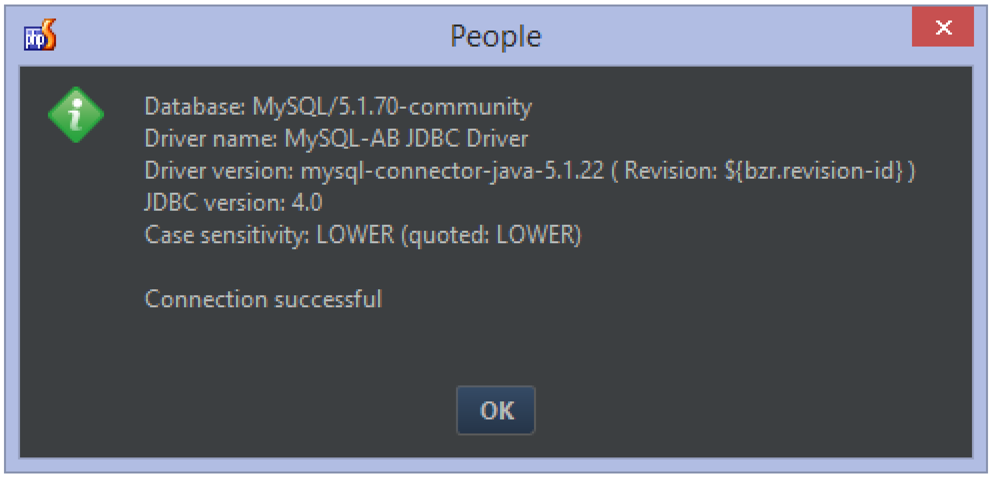
Under the Schemas & Tables tab, we will have to specify which schemas and tables we want to be able to manipulate and generate code completion information for.



	Available databases	Resolve unqualified references
☐	information_schema	☐
✓	people	☐

Table names pattern (java.util.regex.Pattern, empty means .*):

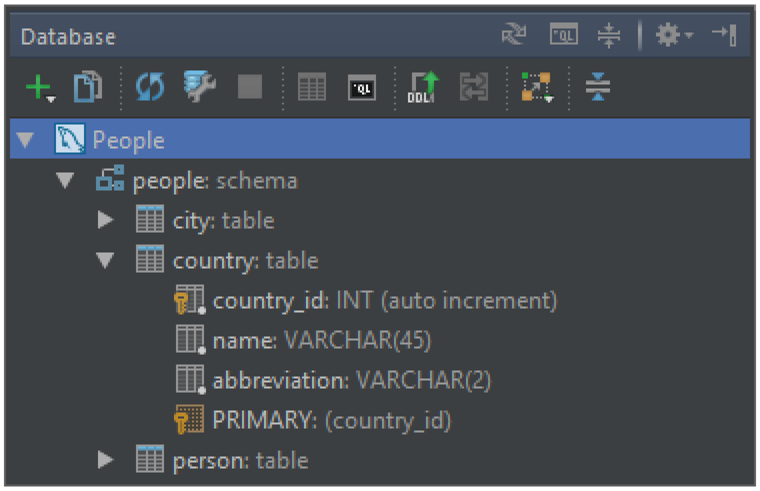
To test the settings, click the Test Connection button, which will tell us if it succeeded (or not) in connecting to the database.



```
Database: MySQL/5.1.70-community
Driver name: MySQL-AB JDBC Driver
Driver version: mysql-connector-java-5.1.22 ( Revision: ${bzr.revision-id} )
JDBC version: 4.0
Case sensitivity: LOWER (quoted: LOWER)

Connection successful
```

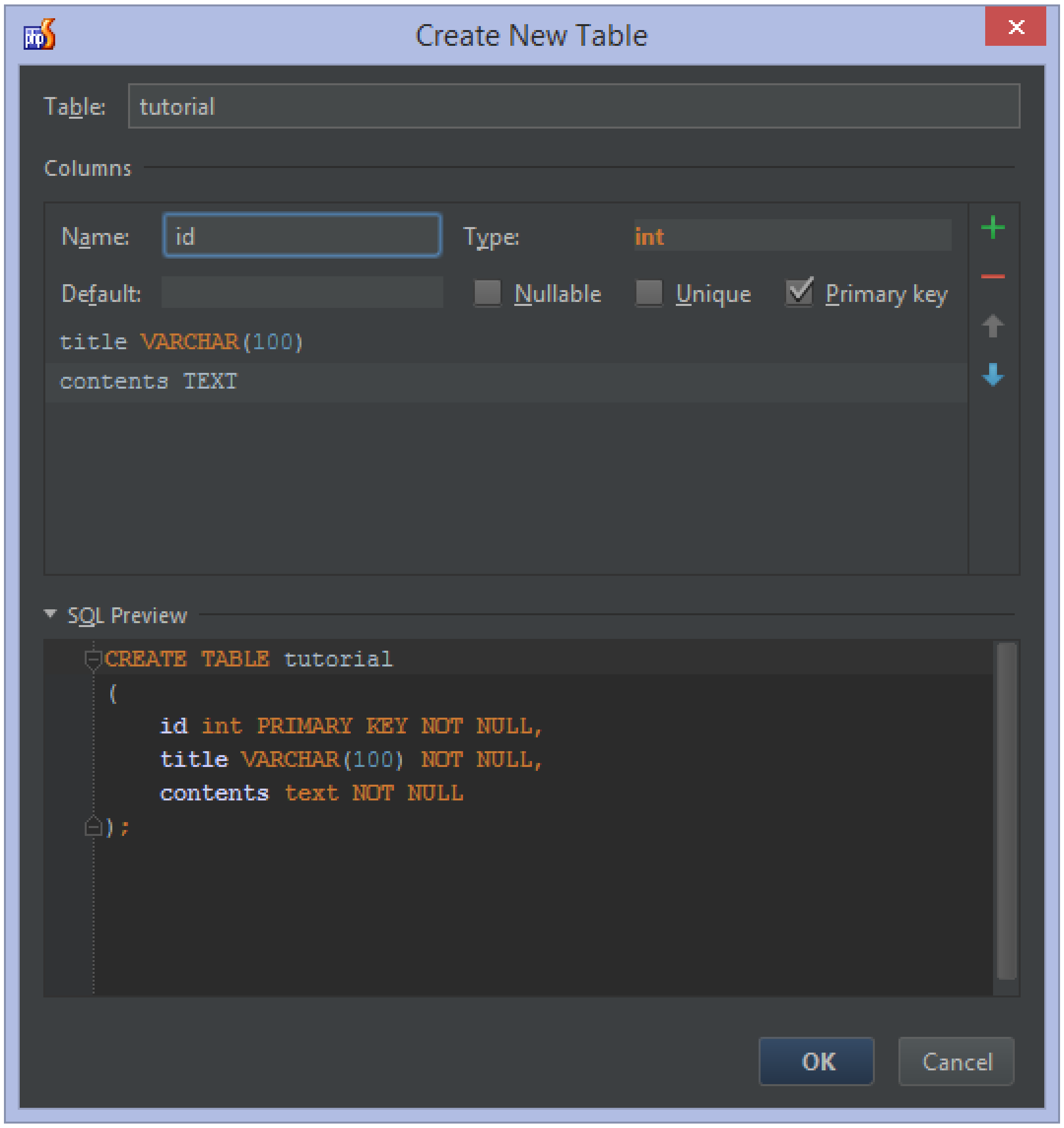
When the connection succeeds, we can save our settings. This will also trigger PhpStorm to download schema information from the database and display it in the database tool window. We can expand schemas, tables, columns and indexes and get more details on the database structure.



Refactoring the Database

Creating a Table

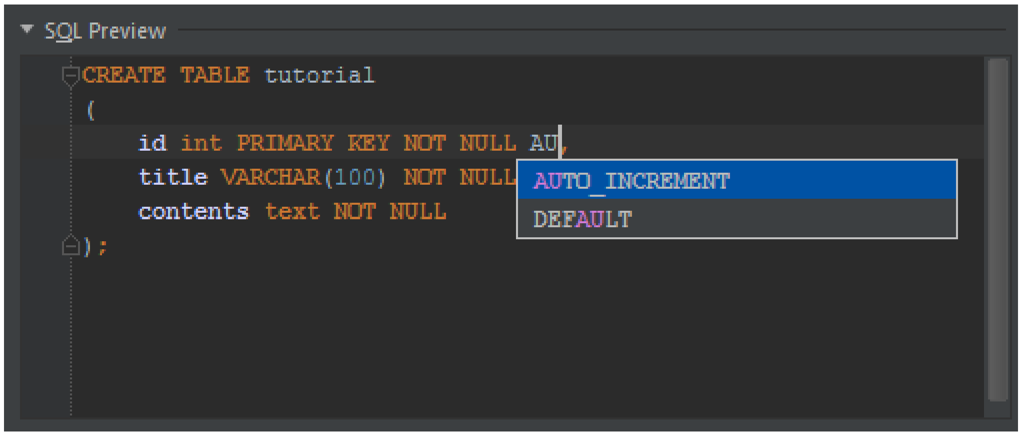
We can use the New | Table context menu (Alt+Insert, or CMD-N on Mac OS X) to create a new table. A dialog will open in which we can give the new table a name and specify the columns that should be created.



```
title VARCHAR(100)
contents TEXT
```

```
CREATE TABLE tutorial
(
    id int PRIMARY KEY NOT NULL,
    title VARCHAR(100) NOT NULL,
    contents text NOT NULL
);
```

Columns can be edited from the UI and PhpStorm will generate the DDL code for us. However, if we wanted to add additional statements to the DDL, we can do so by editing them in the SQL preview editor. Completion is available by pressing Ctrl+Space . Let's make our id column auto increment (in the MySQL dialect).

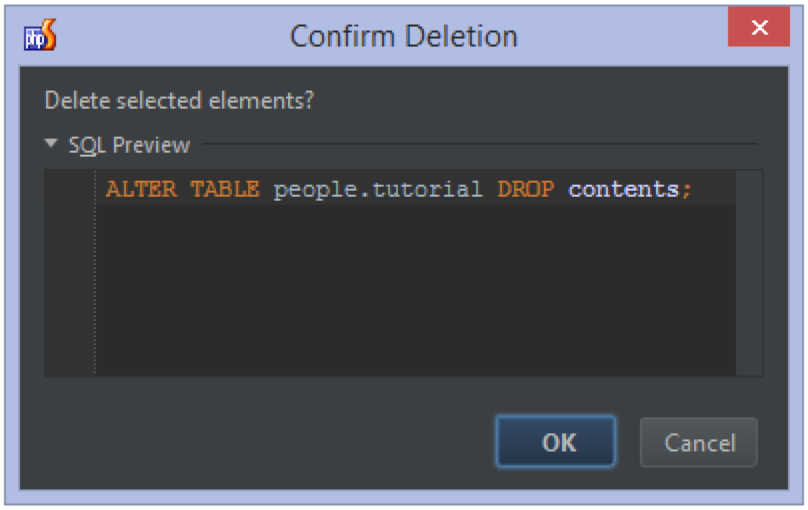


```
CREATE TABLE tutorial
(
    id int PRIMARY KEY NOT NULL AU,
    title VARCHAR(100) NOT NULL,
    contents text NOT NULL
);
```

Clicking OK will create the table in our database.

Deleting a Table, Index or Column

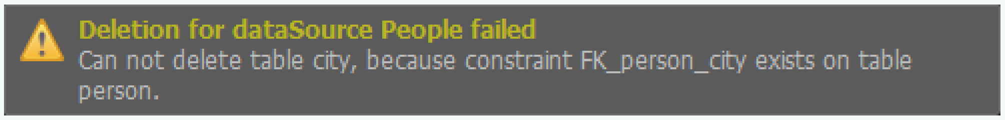
To delete a table, index or column from our database, we can navigate to it and press Delete (or use the Delete context menu). This will show us a preview of the SQL statement that is about to be executed. It could be a DROP TABLE, an ALTER TABLE, or something similar that gets generated.



```
ALTER TABLE people.tutorial DROP contents;
```

Clicking OK will perform the change.

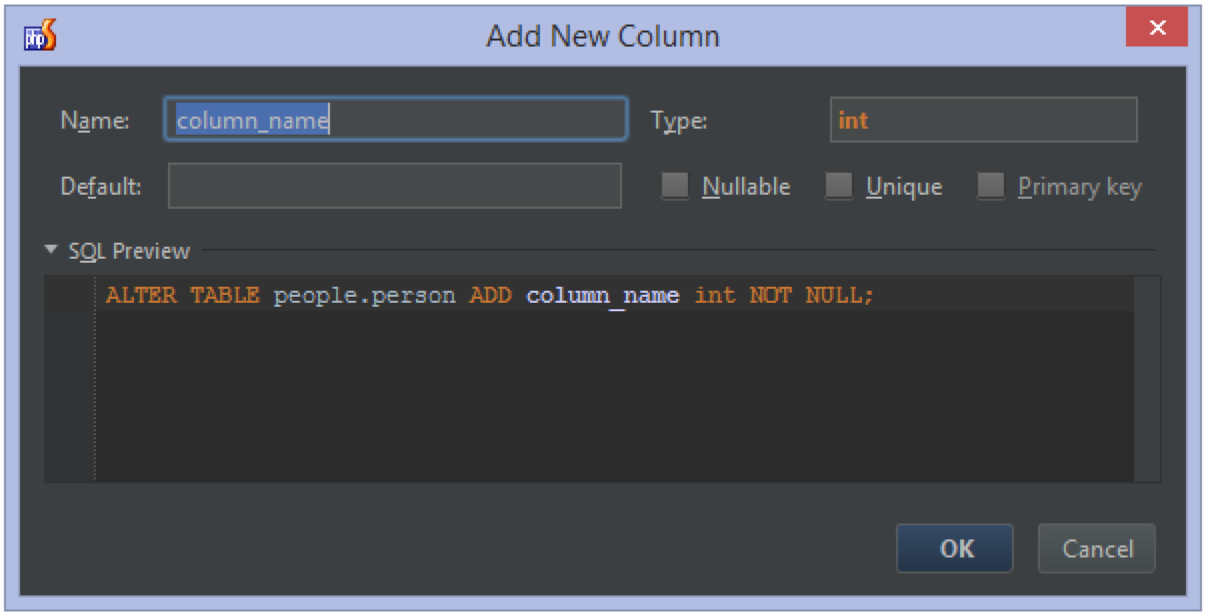
If we attempt to delete a table or column that is referenced in an index or foreign key constraint, the IDE will warn us about this. We will have to manually delete the reference first.



Deletion for dataSource People failed
Can not delete table city, because constraint FK_person_city exists on table person.

Adding a Column to an Existing Table

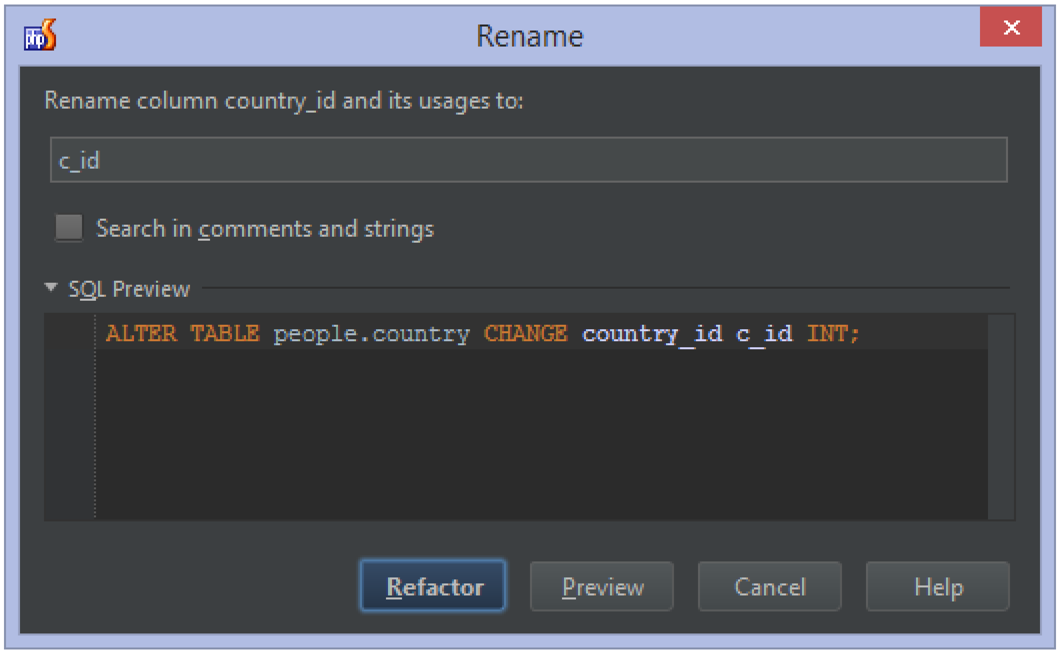
Adding a column to an existing table can be done by selecting the table and using the New | Column context menu or pressing Alt+Insert (CMD-N on Mac OS X). This will give us a UI in which we can generate the ALTER TABLE statement. Clicking OK will apply it to our database.



```
ALTER TABLE people.person ADD column_name int NOT NULL;
```

Renaming a Table, Index or Column

Adding a table, index or column can be done by selecting it and using the Rename refactoring through the Rename... context menu or by pressing Shift+F6. This will let us rename the selected item. We can click Refactor to perform the rename immediately, or first click Preview to get an overview of the changes that will be carried out.

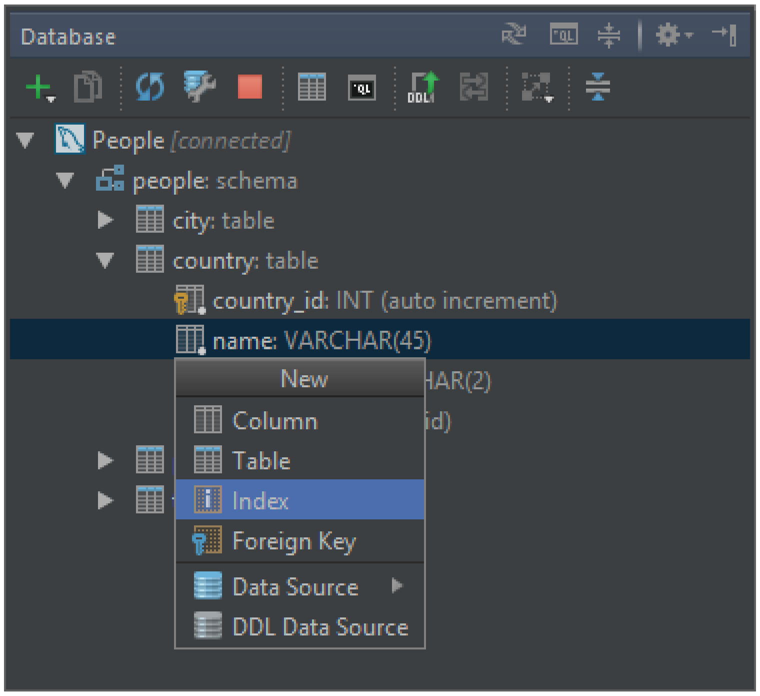

Rename
Rename column country_id and its usages to:
c_id
Search in comments and strings
SQL Preview

```
ALTER TABLE people.country CHANGE country_id c_id INT;
```

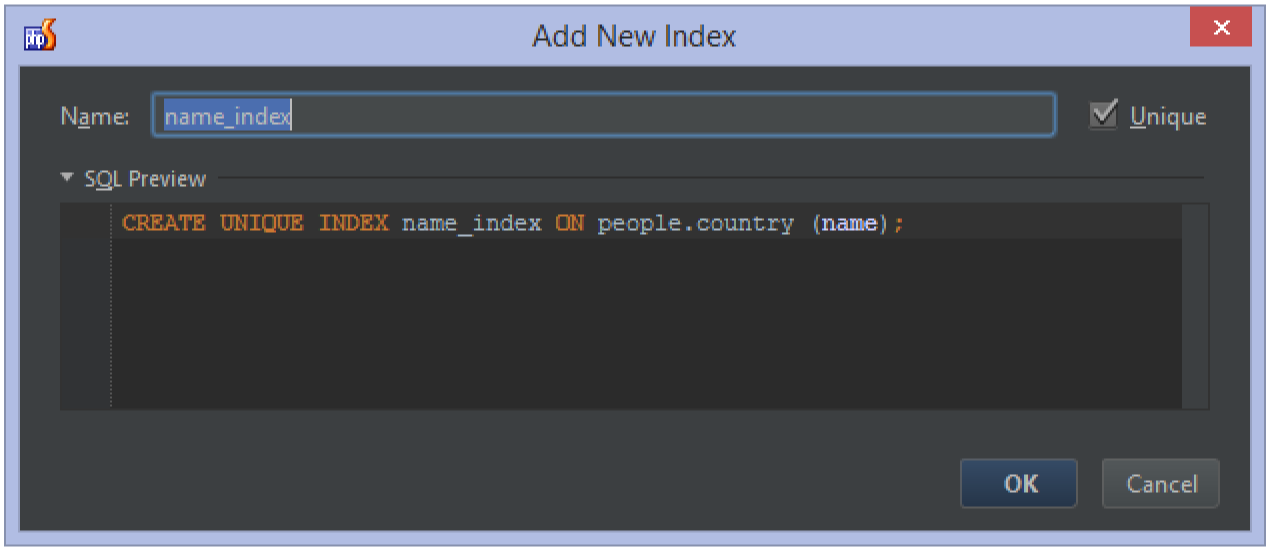
Refactor
Preview
Cancel
Help

Creating Indexes and Foreign Keys

Indexes and foreign key constraints can be added by selecting a column and using the New context menu or pressing Alt+Insert (CMD-N on Mac OS X).



We can name our index and optionally change the generated DDL before clicking OK to add the index or foreign key to the database.

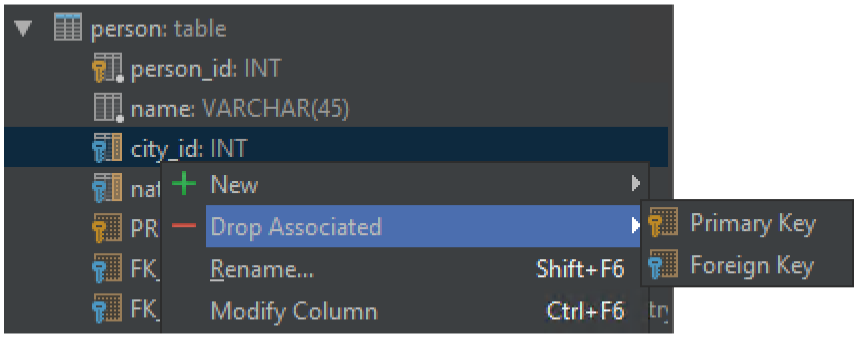


name_index

```
CREATE UNIQUE INDEX name_index ON people.country (name);
```

Deleting Indexes and Foreign Keys

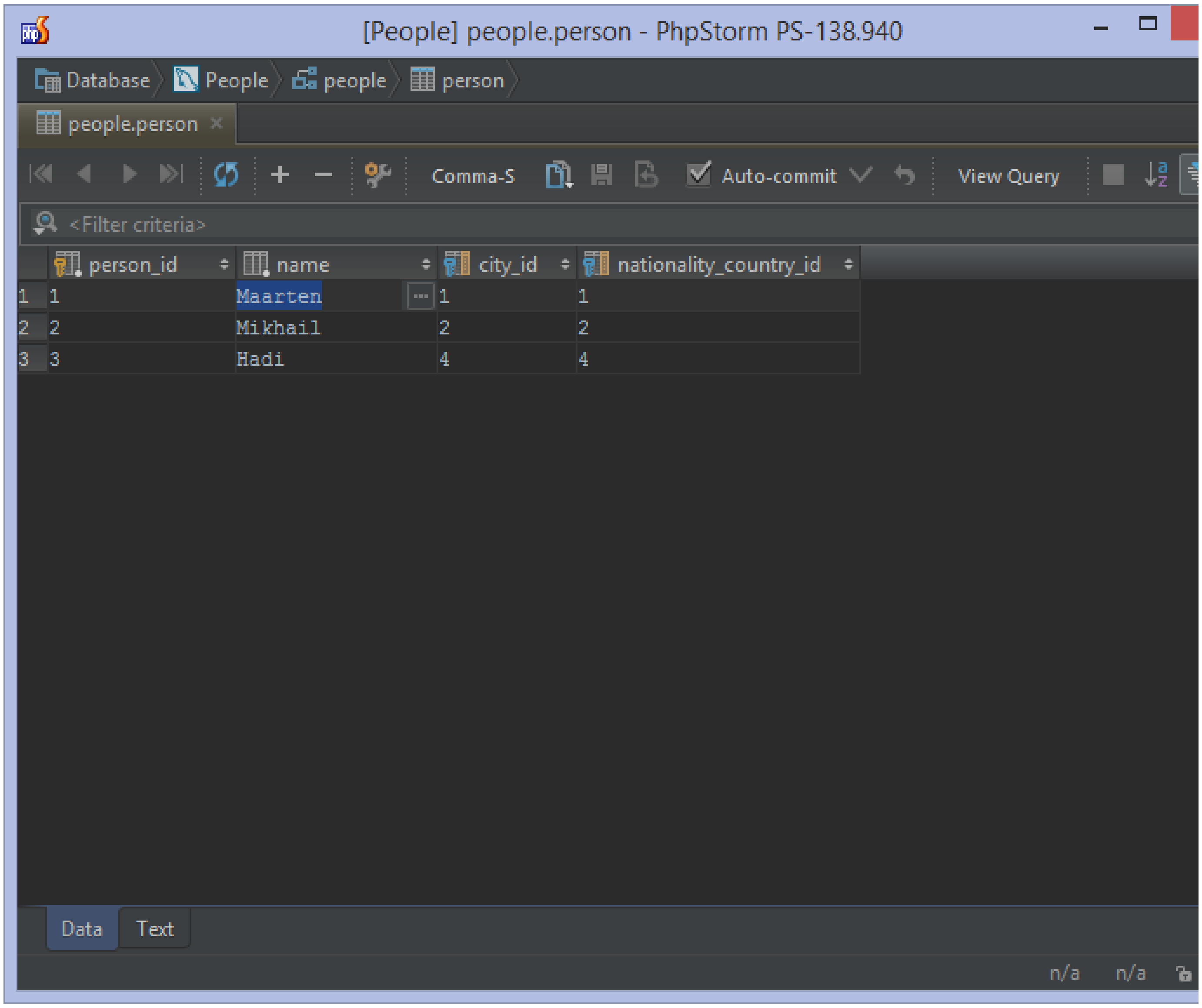
To delete an index or foreign key, we can find it in the database tool window and press Delete. Another option is to find the column for which we want to delete the index or foreign key and use the Drop Associated context menu instead.



Querying, Inserting and Updating data

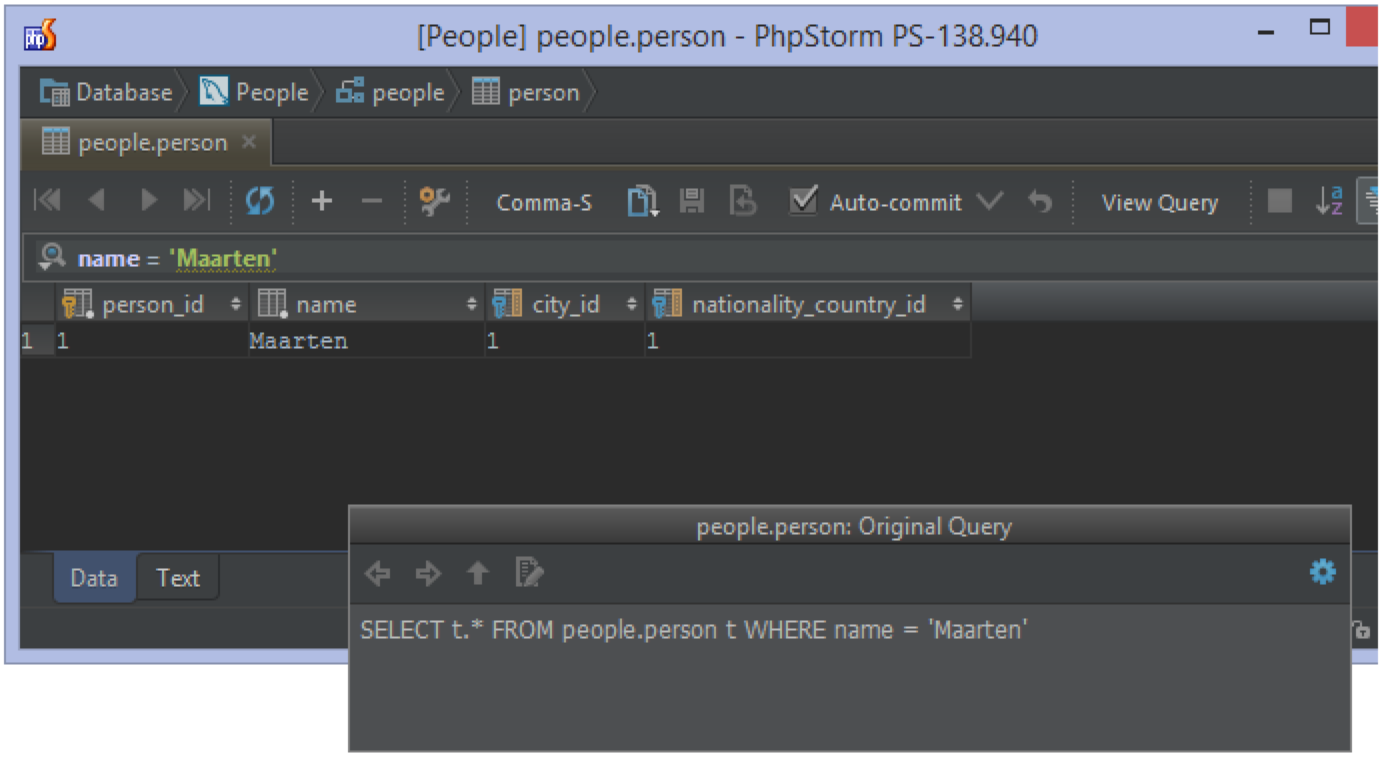
Using the Table Editor

After selecting a table, we can bring up the Table Editor using the context menu or by pressing F4. From the Table Editor, we can view the data in the table and add/edit/remove values. Clicking column headers will sort the data on that column. If the database system supports transactions, we can enable/disable the Auto-commit option. Disabling it will enforce the use of transactions and require us to explicitly commit edits, inserts and deletes.



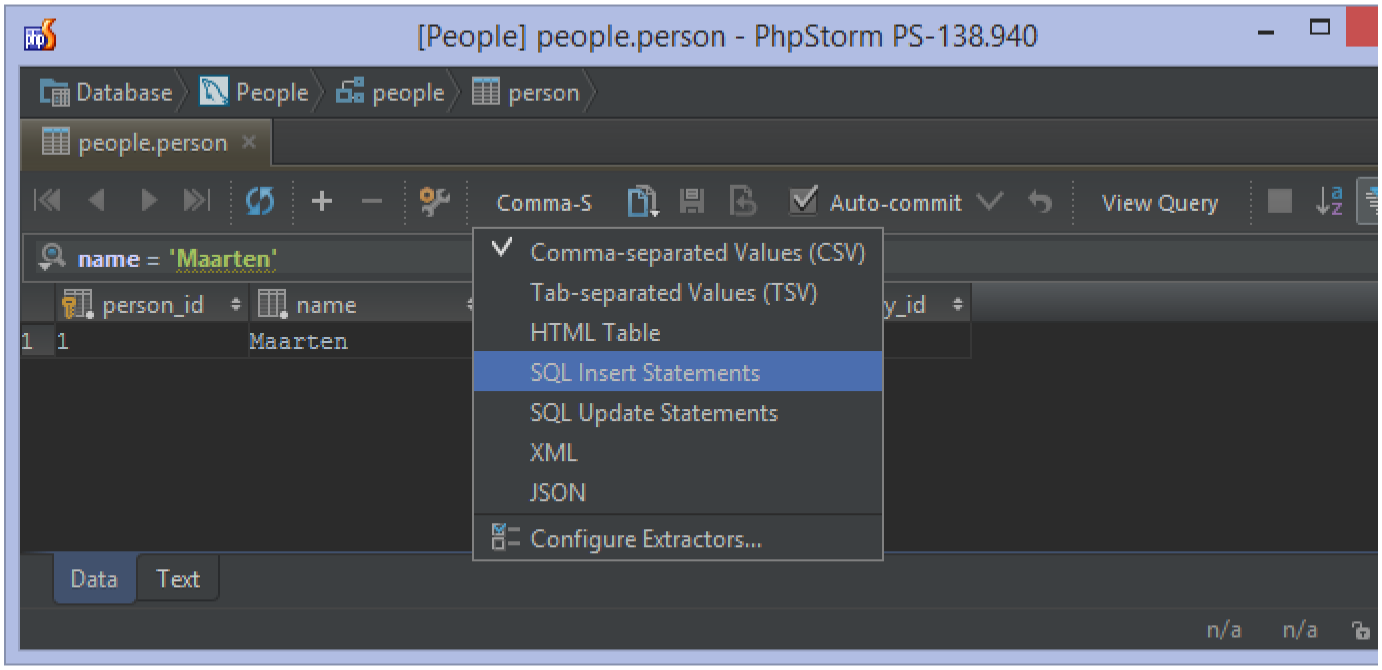
	person_id	name	city_id	nationality_country_id
1	1	Maarten	1	1
2	2	Mikhail	2	2
3	3	Hadi	4	4

We can also filter data by adding a filter statement (use Ctrl+Space for completion on columns). Clicking View Query will show the resulting query.



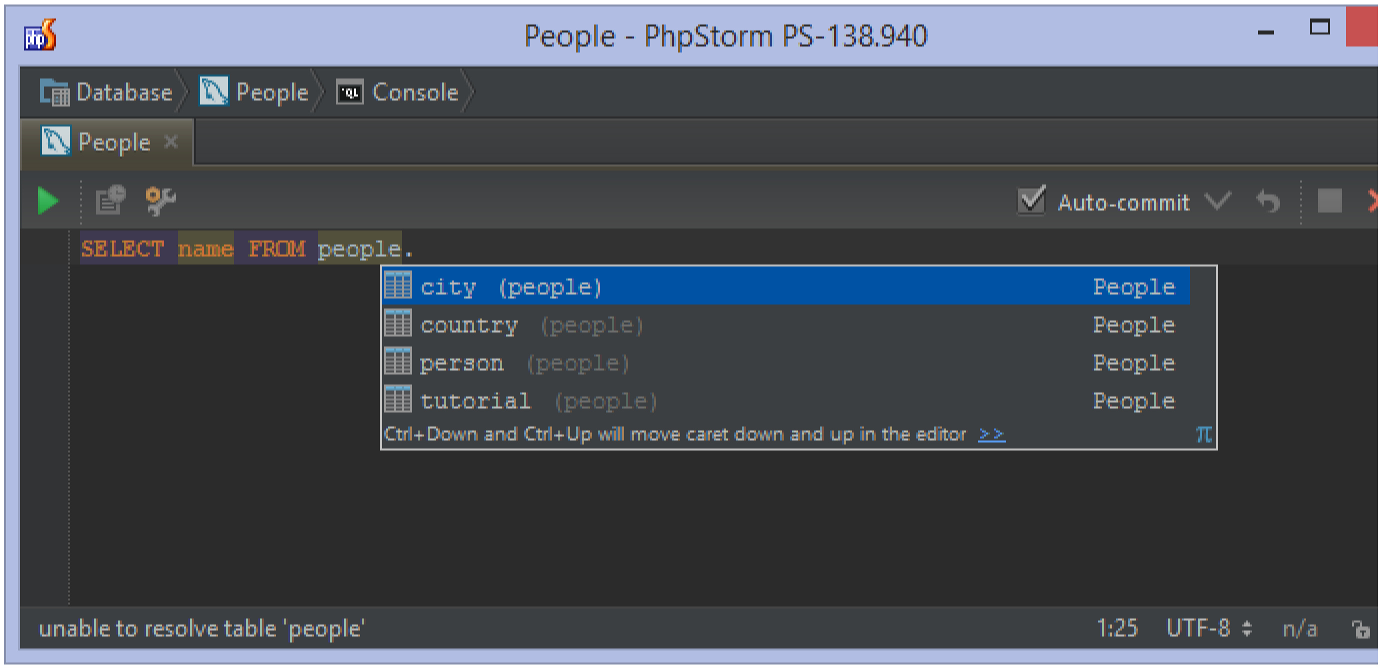
```
SELECT t.* FROM people.person t WHERE name = 'Maarten'
```

The Table Editor as well as the Database Console support exporting data to the clipboard or a file. From the toolbar, we can select the desired output format, such as Comma-Separated Values (CSV), XML or JSON. We can also generate INSERT and UPDATE statements from the data so we can easily import it into a different database afterwards.



Using the Database Console

While the Table Editor gives us a number of options for working with data, it's limited to just one table and does not allow us to run an arbitrary SQL statement. The Database Console is a query editor in which we can write any SQL statement and run it. We can open the Database Console by pressing Ctrl+Shift+F10 (Shift-CMD-F10 on Mac OS X).



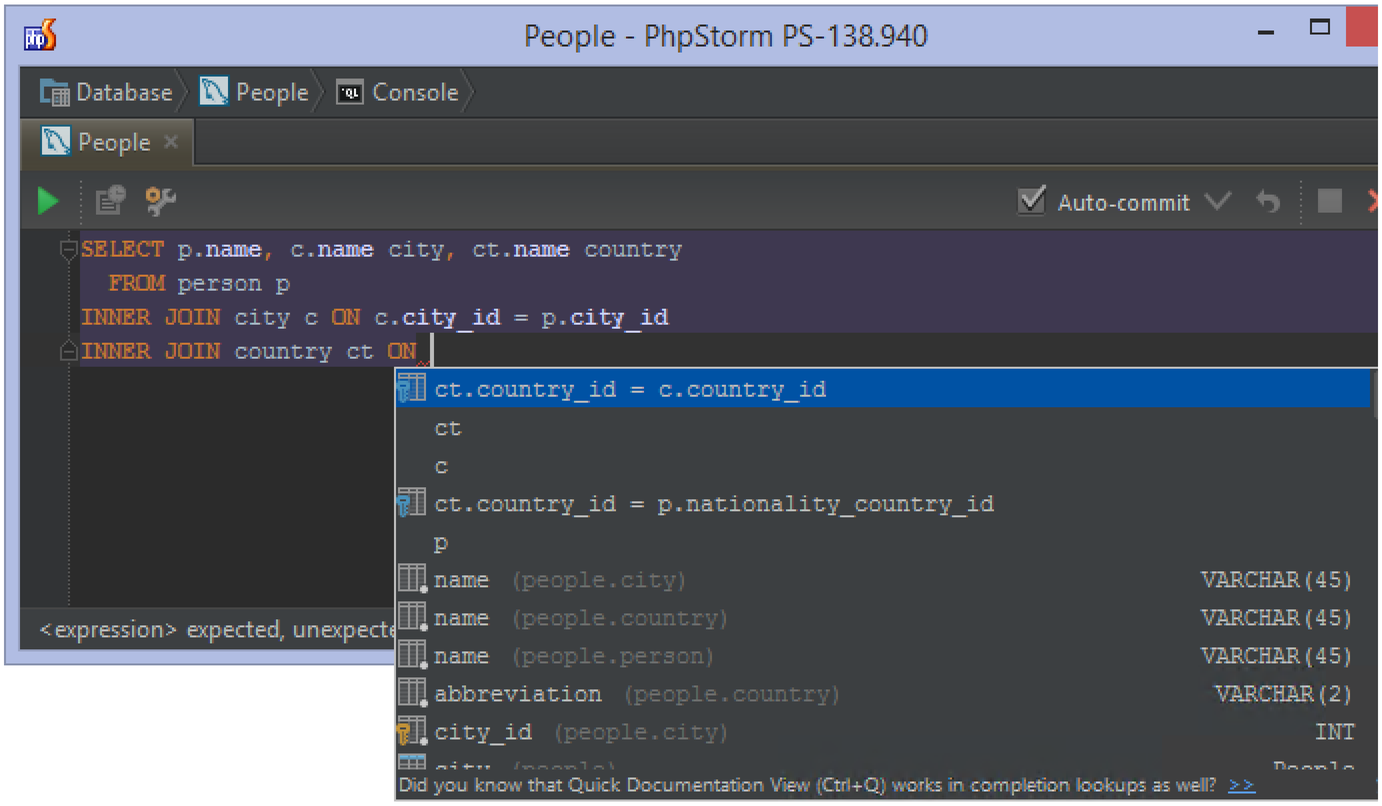
People - PhpStorm PS-138.940

```
SELECT name FROM people.
```

```
city  (people)          People
country  (people)       People
person  (people)        People
tutorial  (people)      People
Ctrl+Down and Ctrl+Up will move caret down and up in the editor >>
```

unable to resolve table 'people'   1:25   UTF-8   n/a

We can enter a query in the console and get syntax highlighting, completion (Ctrl+Space), on-the-fly code analysis, multiple intentions, and navigation. Completion can even figure out foreign key constraints and suggest JOIN conditions.



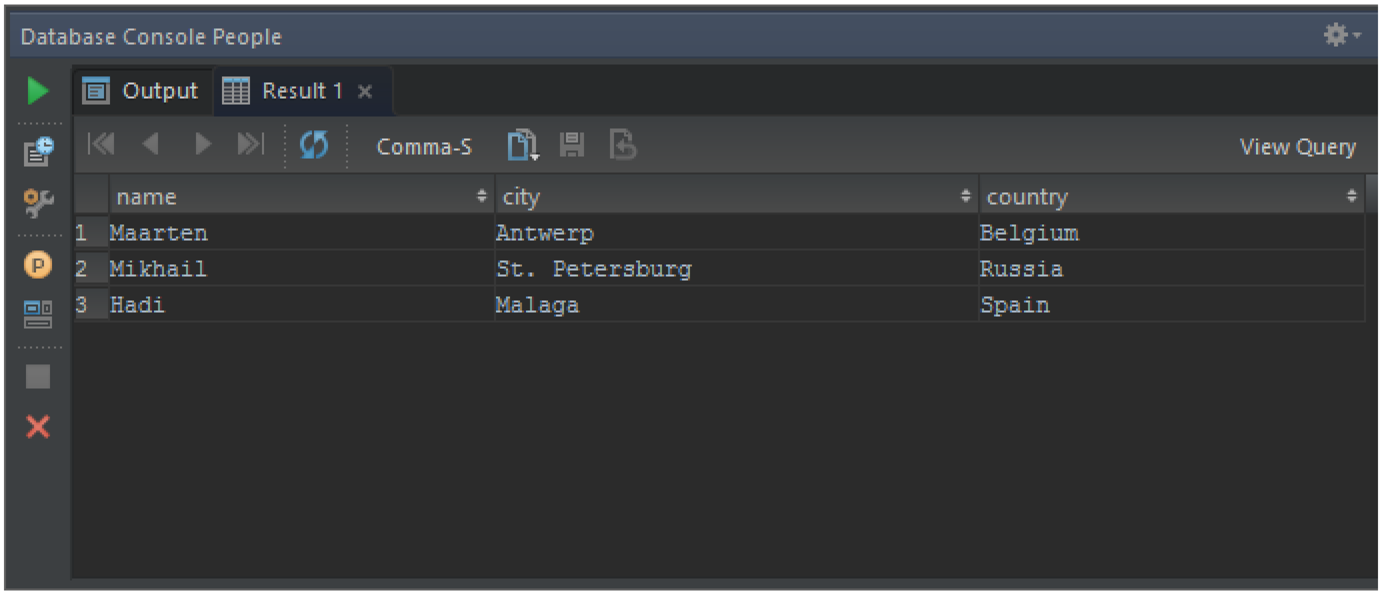
People - PhpStorm PS-138.940

```
SELECT p.name, c.name city, ct.name country
  FROM person p
INNER JOIN city c ON c.city_id = p.city_id
INNER JOIN country ct ON 
```

```
ct.country_id = c.country_id
ct
c
ct.country_id = p.nationality_country_id
p
name  (people.city)                 VARCHAR(45)
name  (people.country)              VARCHAR(45)
name  (people.person)               VARCHAR(45)
abbreviation  (people.country)      VARCHAR(2)
city_id  (people.city)              INT
Did you know that Quick Documentation View (Ctrl+Q) works in completion lookups as well? >>
```

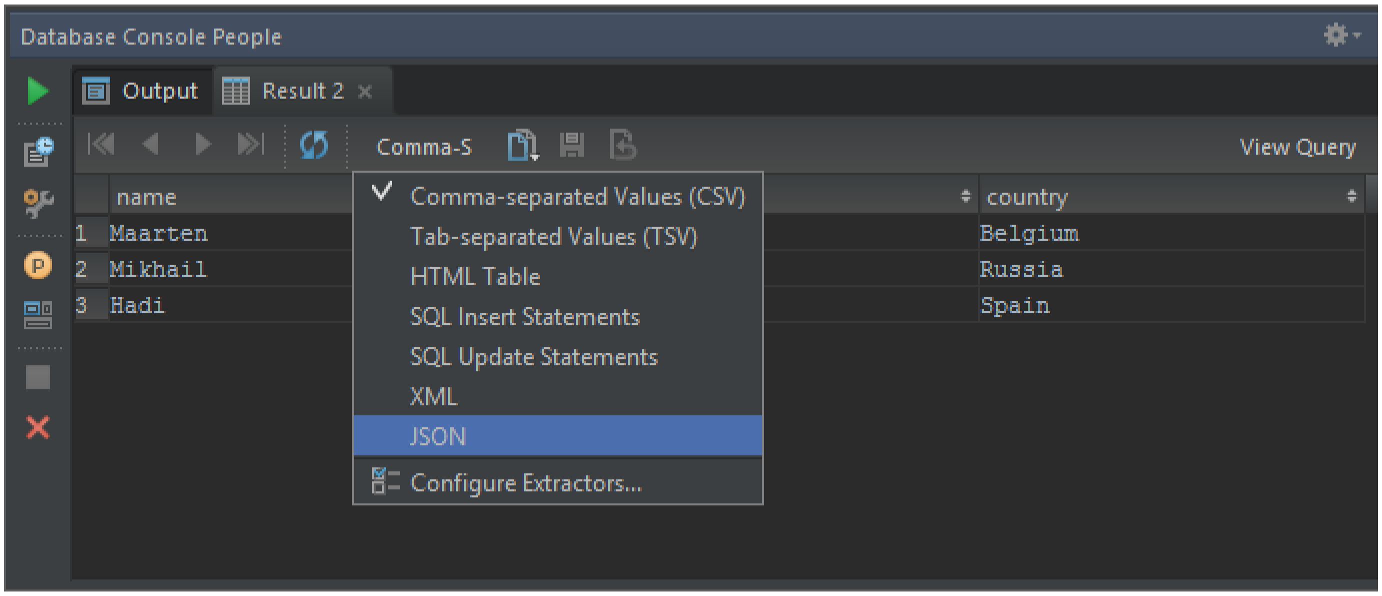
<expression> expected, unexpecte

To run the query, we select the statement and press Ctrl+Enter. The result set (or query log) is displayed in a separate tool window.



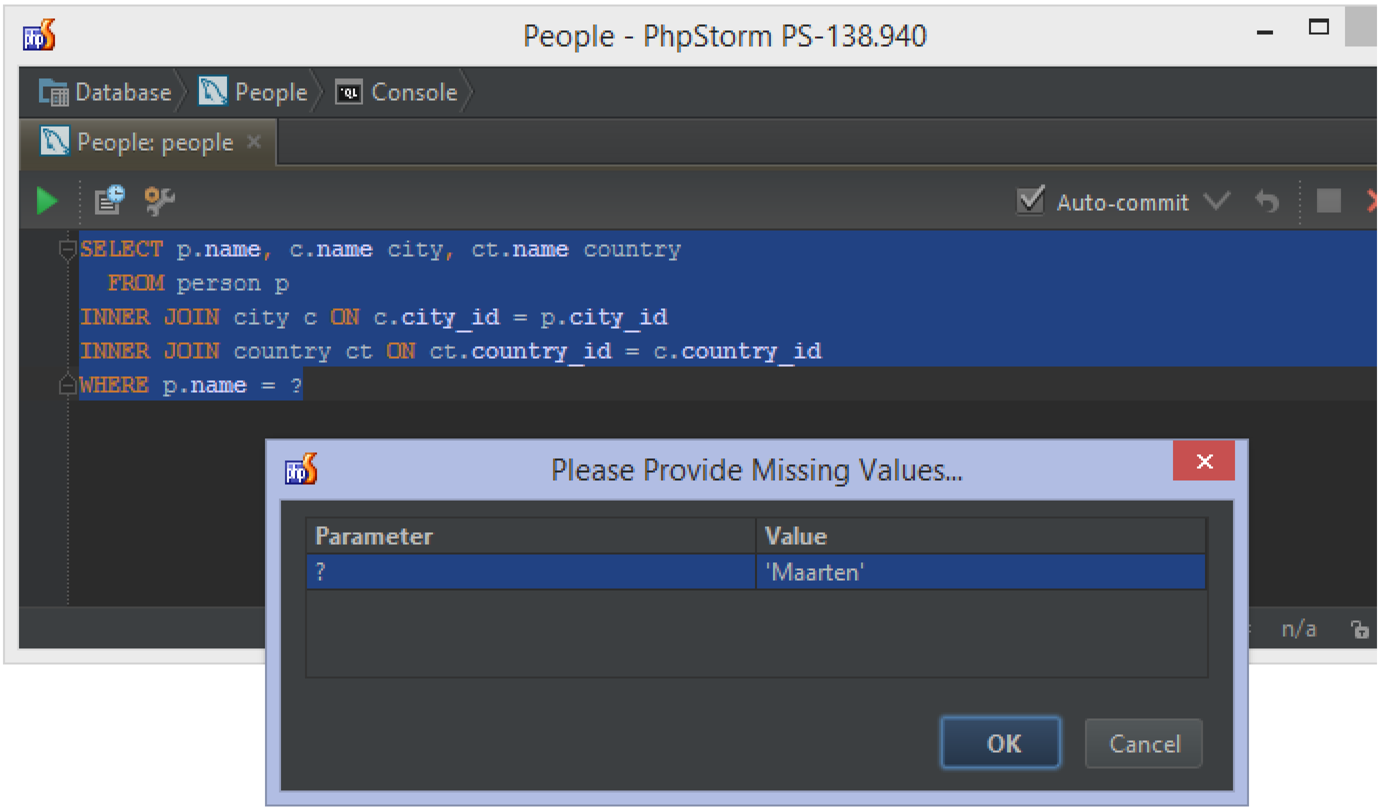
	name	city	country
1	Maarten	Antwerp	Belgium
2	Mikhail	St. Petersburg	Russia
3	Hadi	Malaga	Spain

The Table Editor as well as the Database Console support exporting data to the clipboard or a file. From the toolbar, we can select the desired output format, such as Comma-Separated Values (CSV), XML or JSON. We can also generate INSERT and UPDATE statements from the data so we can easily import it into a different database afterwards.



	name	country
1	Maarten	Belgium
2	Mikhail	Russia
3	Hadi	Spain

When building queries in the Database Console, we can make use of variables and substitute them when running the query. By default, variables are recognized as strings similar to ?, :variable, @variable, #variable# or $variable$. By using variables, PhpStorm will prompt for the values to use in the query during execution.


People - PhpStorm PS-138.940

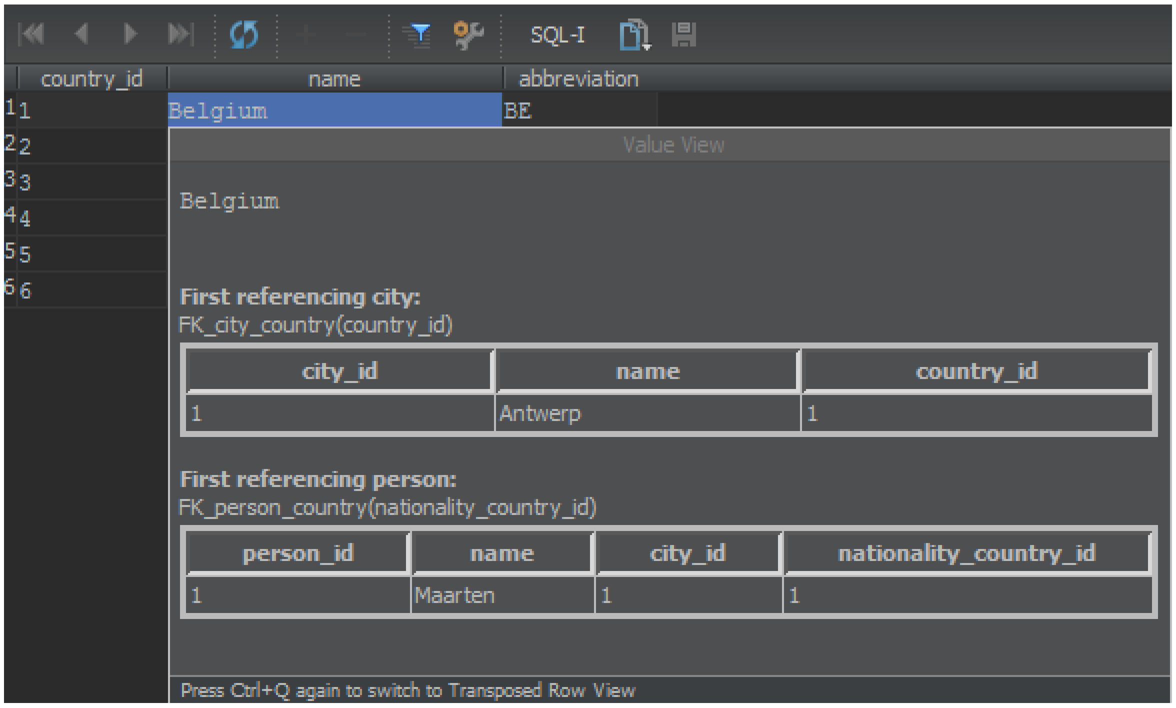
```
SELECT p.name, c.name city, ct.name country
  FROM person p
INNER JOIN city c ON c.city_id = p.city_id
INNER JOIN country ct ON ct.country_id = c.country_id
WHERE p.name = ?
```

Please Provide Missing Values...

Parameter	Value
?	'Maarten'

Value View / Single Record Transpose View

When browsing a table, we may want to see an example of where the selected value is referenced. Rather than generating a database diagram, we can instead press Ctrl+Q (Ctrl-J on Mac OS X) and get an example of how a given row is used throughout the database.



	country_id	name	abbreviation
1	1	Belgium	BE
2	2		
3	3		
4	4		
5	5		
6	6		

Value View

Belgium

First referencing city:
FK_city_country(country_id)

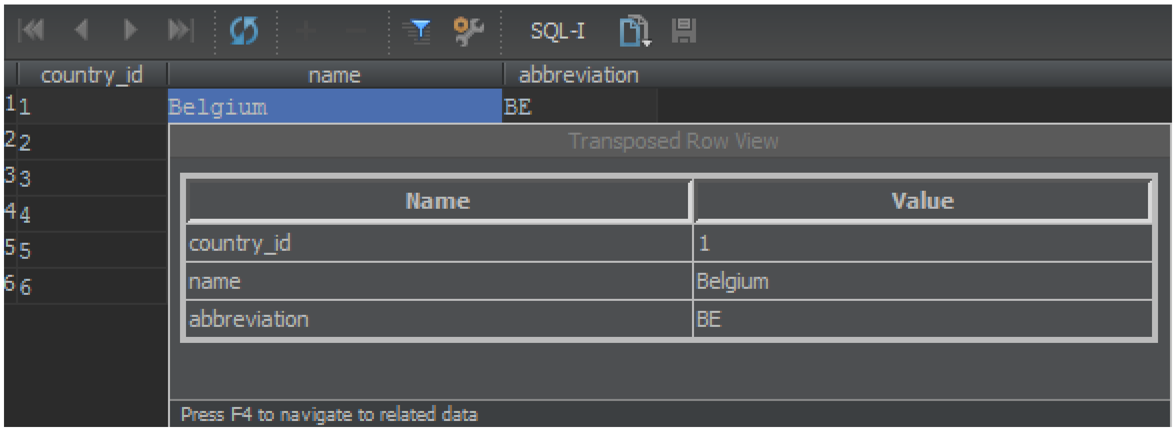
city_id	name	country_id
1	Antwerp	1

First referencing person:
FK_person_country(nationality_country_id)

person_id	name	city_id	nationality_country_id
1	Maarten	1	1

Press Ctrl+Q again to switch to Transposed Row View

If we press the keyboard shortcut again, we get a transposed view of the selected row. When we work with a table that has many fields and requires vertical scrolling, the Single Record Transpose View makes it easier to view the data by flipping the column and row axis.



SQL-I

country_id	name	abbreviation
1	Belgium	BE
2		
3		
4		
5		
6		

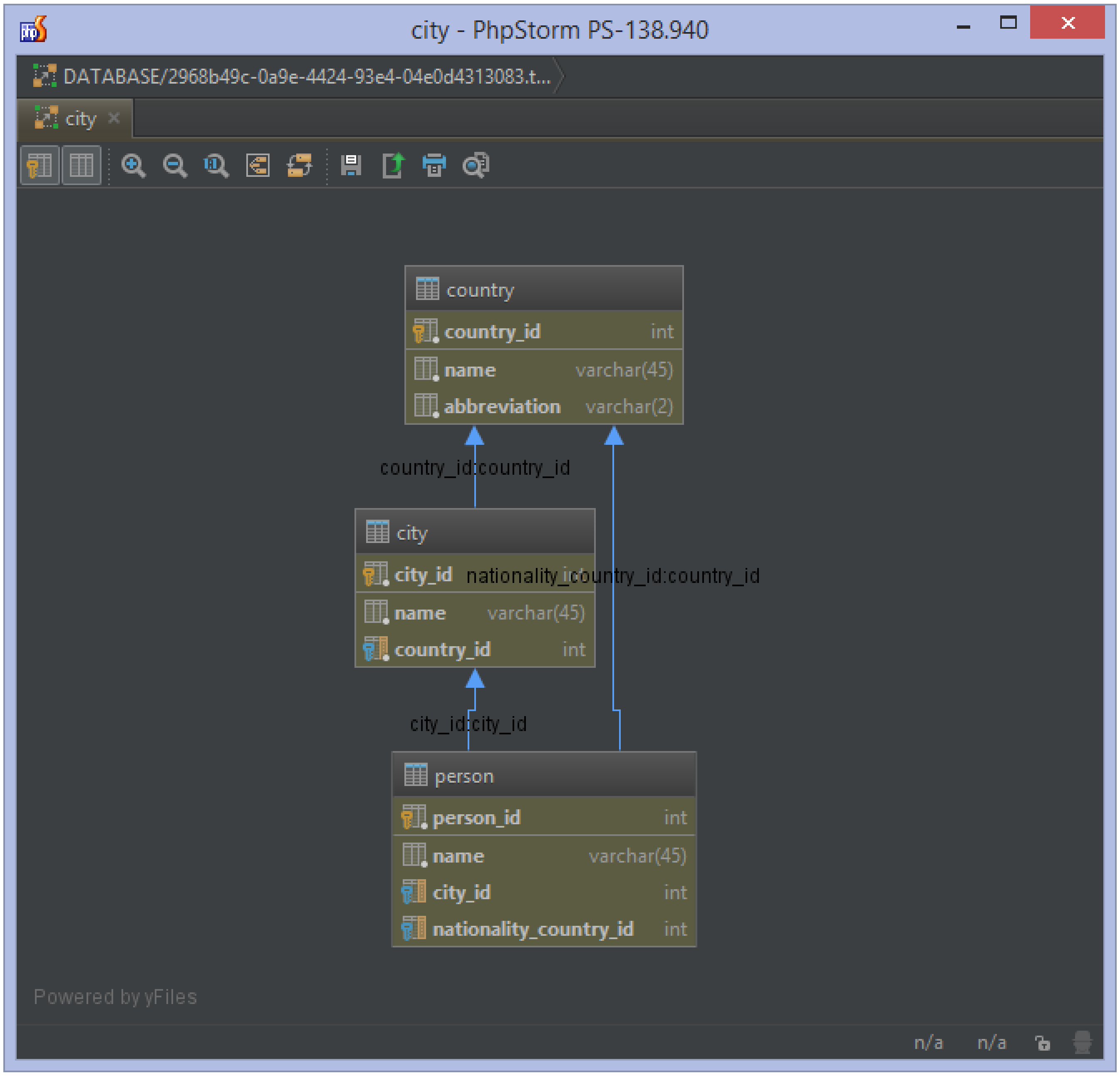
Transposed Row View

Name	Value
country_id	1
name	Belgium
abbreviation	BE

Press F4 to navigate to related data

Generating a UML database diagram

If we have multiple tables with foreign keys between them, we can easily generate a database diagram. Select the tables to be included in the diagram and use the Diagrams | Show Visualization... context menu or press Ctrl+Alt+Shift+U (Alt-Shift-CMD-U on Mac OS X). PhpStorm will then generate a database diagram for these tables, showing how they relate to each other.



We can export the diagram to a file, add annotations to it, or print it.

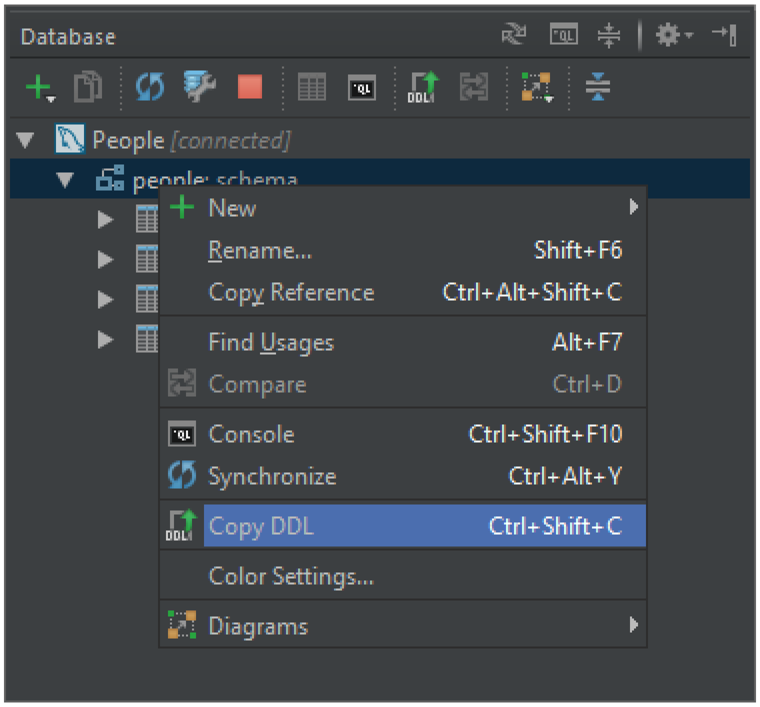
If all we need is a quick overview of the database, Ctrl+Alt+U (Alt-CMD-U on Mac OS X) will show the diagram in a popup instead.

Database Migration

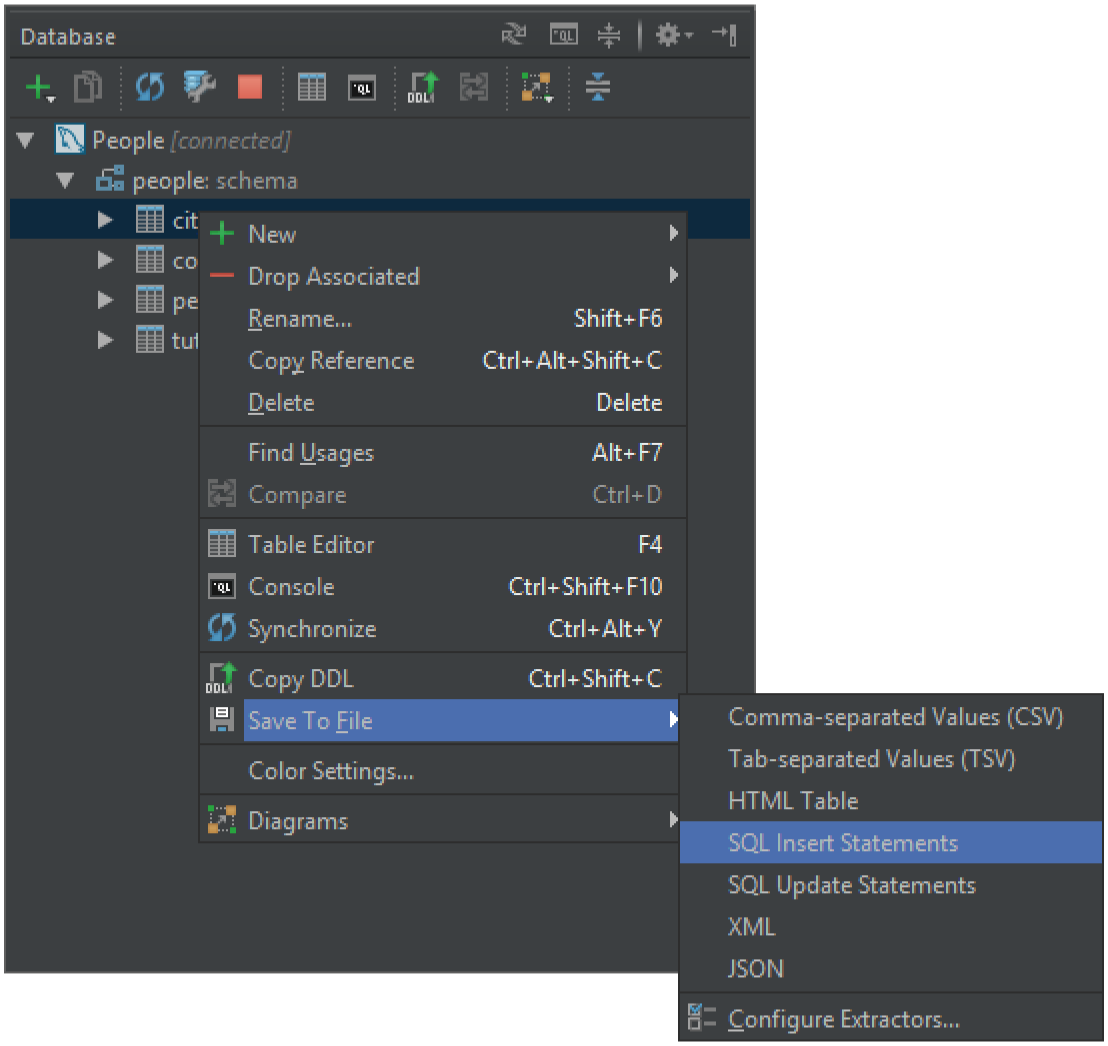
Exporting Database Tables as SQL Statements

Often, we want to be able to export databases so we can import them in other databases. For example: copying the schema of a newly created table on our development machine to a test server, or vice versa.

PhpStorm allows exporting tables as (DDL) SQL statements: using the Copy DDL context menu on one or more tables scripts the tables, columns, indexes and foreign keys to the clipboard so we can paste them in database console.

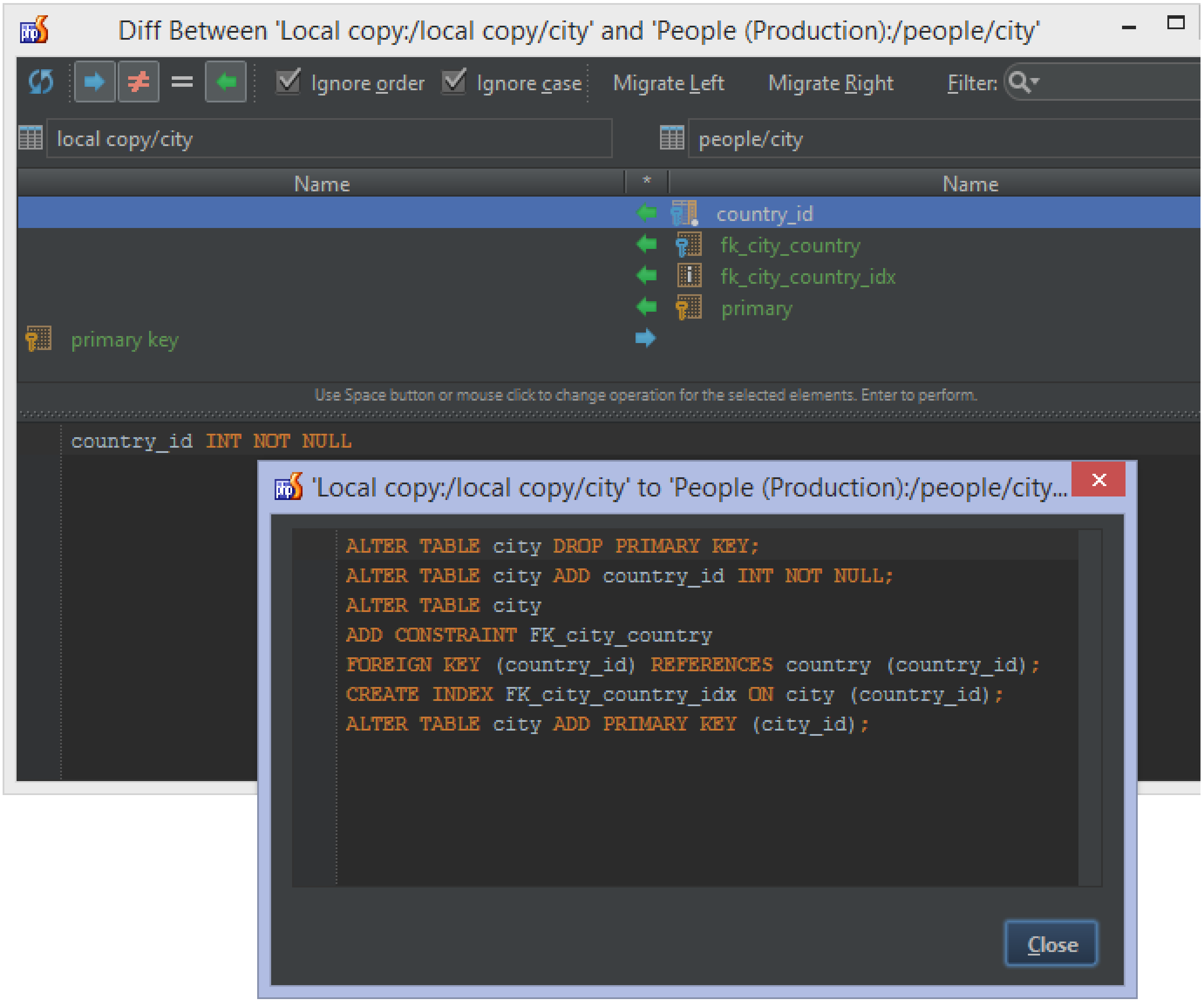


If we also need the contents of the various tables, we can use the Save To File | SQL Insert Statements context menu and export the data of the table(s) as well.



Comparing and Migrating Database Code

When we have several data sources configured, for example a production database and a local copy, we can select both of them (or just select two specific tables in each) and use the Compare context menu (Ctrl+D, or CMD-D on Mac OS X). This will generate a "diff" of both data sources or tables and allows us generate a migration script.


Diff Between 'Local copy:/local copy/city' and 'People (Production):/people/city'
Ignore order
Ignore case
Migrate Left
Migrate Right
Filter:
local copy/city
people/city
Name
*
Name
country_id
fk_city_country
fk_city_country_idx
primary
primary key
Use Space button or mouse click to change operation for the selected elements. Enter to perform.
'Local copy:/local copy/city' to 'People (Production):/people/city...
Close

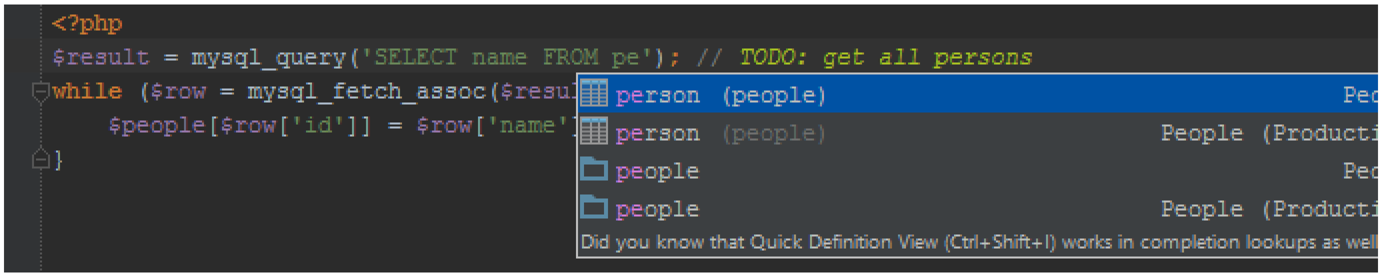
```
country_id INT NOT NULL
```

```
ALTER TABLE city DROP PRIMARY KEY;
ALTER TABLE city ADD country_id INT NOT NULL;
ALTER TABLE city
ADD CONSTRAINT FK_city_country
FOREIGN KEY (country_id) REFERENCES country (country_id);
CREATE INDEX FK_city_country_idx ON city (country_id);
ALTER TABLE city ADD PRIMARY KEY (city_id);
```

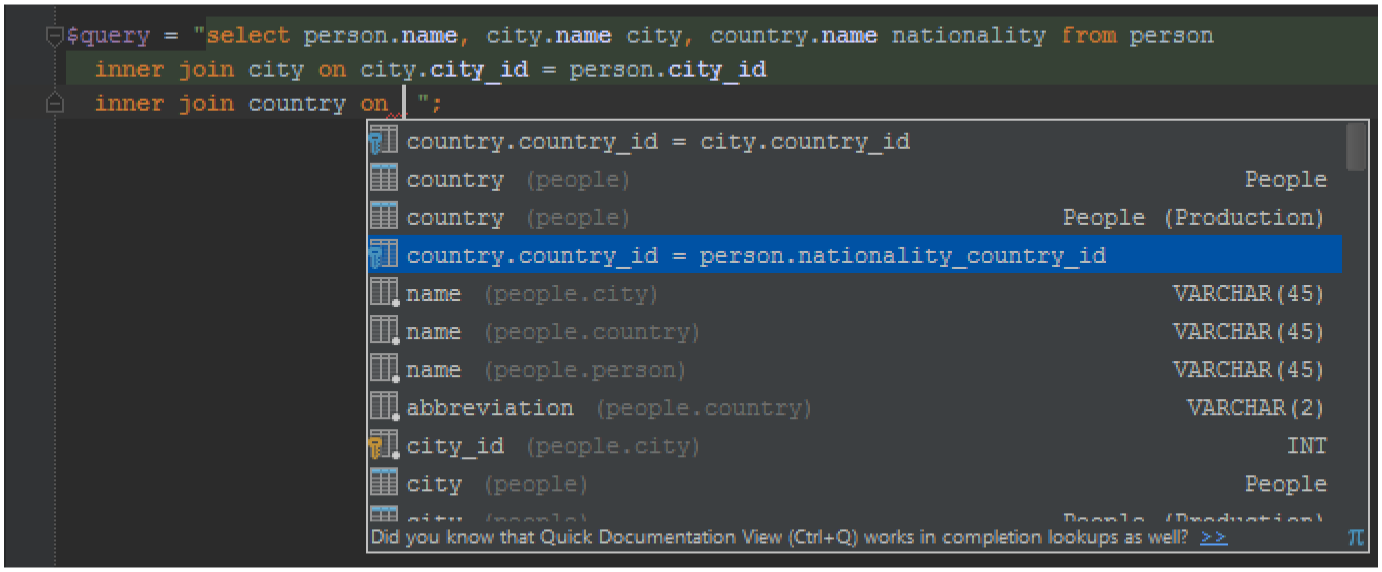
Above, we can see our local database table does not have the country_id column and related primary and foreign keys. Clicking Migrate Left will generate a script that would bring our local database table to the latest version. Migrate Right would do the opposite.

Database Code Completion while working in PHP

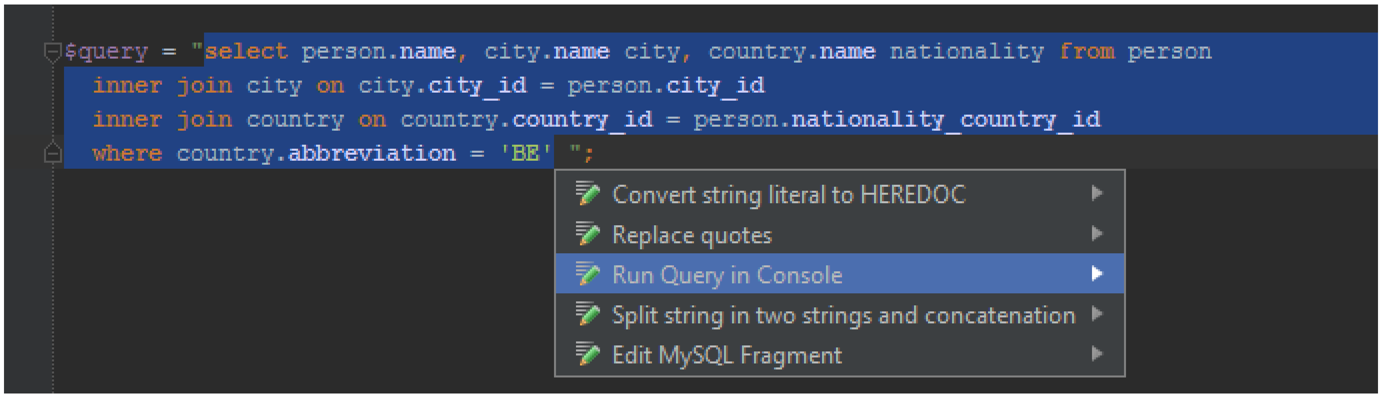
While writing our code, we can get all the features of the Database Console inside PHP, JavaScript, HTML or other languages. We can just start writing our SQL statement and after a while (typically right after the FROM), PhpStorm will provide us with code completion (Ctrl+Space). If completion is required earlier, for example to be able to complete column names, we can place the cursor in the string literal, write "SELECT", press Alt+Enter , and select the language to use (MySQL, for example).


```
<?php
$result = mysql_query('SELECT name FROM pe'); // TODO: get all persons
while ($row = mysql_fetch_assoc($resu
    $people[$row['id']] = $row['name']
}
```
```
person  (people)                    Pe
person  (people)     People  (Producti
people                              Pe
people               People  (Producti
Did you know that Quick Definition View (Ctrl+Shift+I) works in completion lookups as well
```

We will get completion on JOIN statements, too!

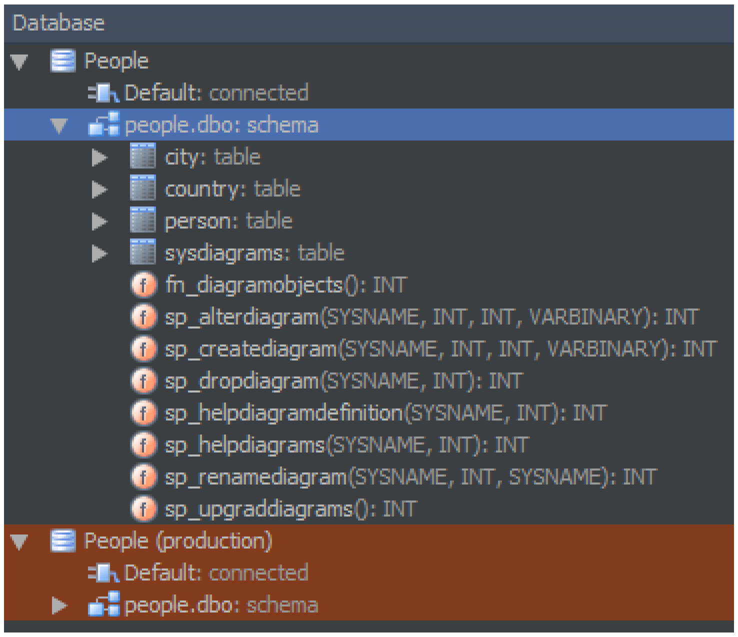

```
$query = "select person.name, city.name city, country.name nationality from person
  inner join city on city.city_id = person.city_id
  inner join country on  ";
```
```
country.country_id = city.country_id
country  (people)                              People
country  (people)                  People  (Production)
country.country_id = person.nationality_country_id
name  (people.city)                         VARCHAR(45)
name  (people.country)                      VARCHAR(45)
name  (people.person)                       VARCHAR(45)
abbreviation  (people.country)               VARCHAR(2)
city_id  (people.city)                              INT
city  (people)                                   People
Did you know that Quick Documentation View (Ctrl+Q) works in completion lookups as well? >>
```

After selecting (part of) a query, we can press Alt+Enter+* and edit the MySQL fragment, or pick the Run Query in Console ent ry to do exactly that: copy the query into a Database Console and run it there.

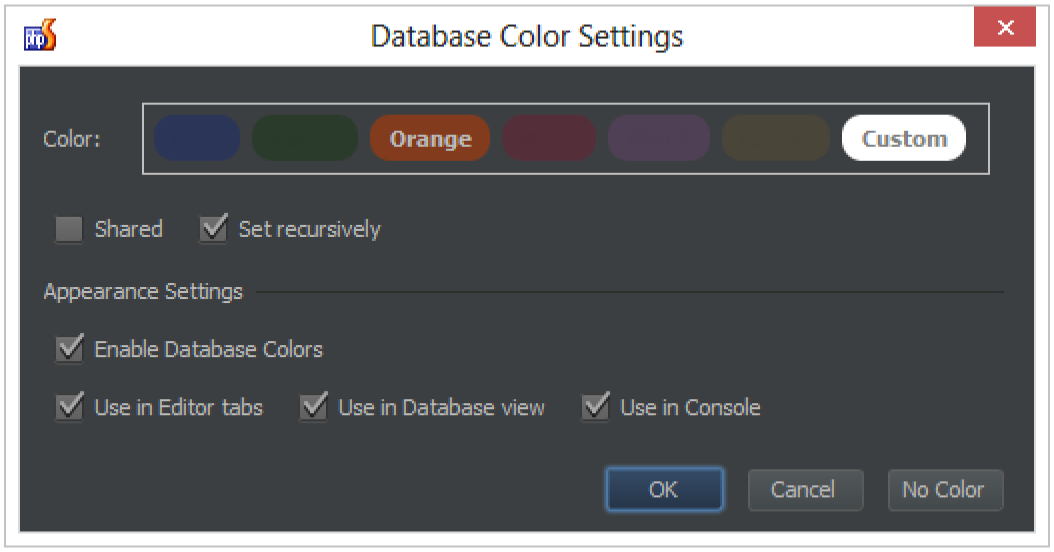

```
$query = "select person.name, city.name city, country.name nationality from person
  inner join city on city.city_id = person.city_id
  inner join country on country.country_id = person.nationality_country_id
  where country.abbreviation = 'BE' ";
```
Convert string literal to HEREDOC
Replace quotes
Run Query in Console
Split string in two strings and concatenation
Edit MySQL Fragment

Coloring Data Sources

When working with multiple data sources or data sources that have multiple schemas, we can assign different colors to each of them. For example, when working with a development and production database at the same time, it can be useful to give both a different color to have a visual clue about the database we are currently working with.


```
Database
▼ People
    Default: connected
  ▼ people.dbo: schema
    ▶ city: table
    ▶ country: table
    ▶ person: table
    ▶ sysdiagrams: table
      fn_diagramobjects(): INT
      sp_alterdiagram(SYSNAME, INT, INT, VARBINARY): INT
      sp_creatediagram(SYSNAME, INT, INT, VARBINARY): INT
      sp_dropdiagram(SYSNAME, INT): INT
      sp_helpdiagramdefinition(SYSNAME, INT): INT
      sp_helpdiagrams(SYSNAME, INT): INT
      sp_renamediagram(SYSNAME, INT, SYSNAME): INT
      sp_upgraddiagrams(): INT
▼ People (production)
    Default: connected
  ▶ people.dbo: schema
```

Colors can be enabled and assigned by using the Color Settings context menu on a data source, schema or object.


Database Color Settings

Color: Orange | Custom

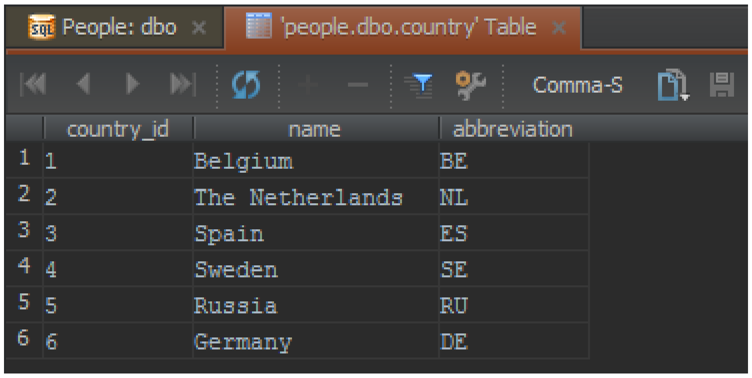
[ ] Shared [✓] Set recursively

Appearance Settings

[✓] Enable Database Colors

[✓] Use in Editor tabs [✓] Use in Database view [✓] Use in Console

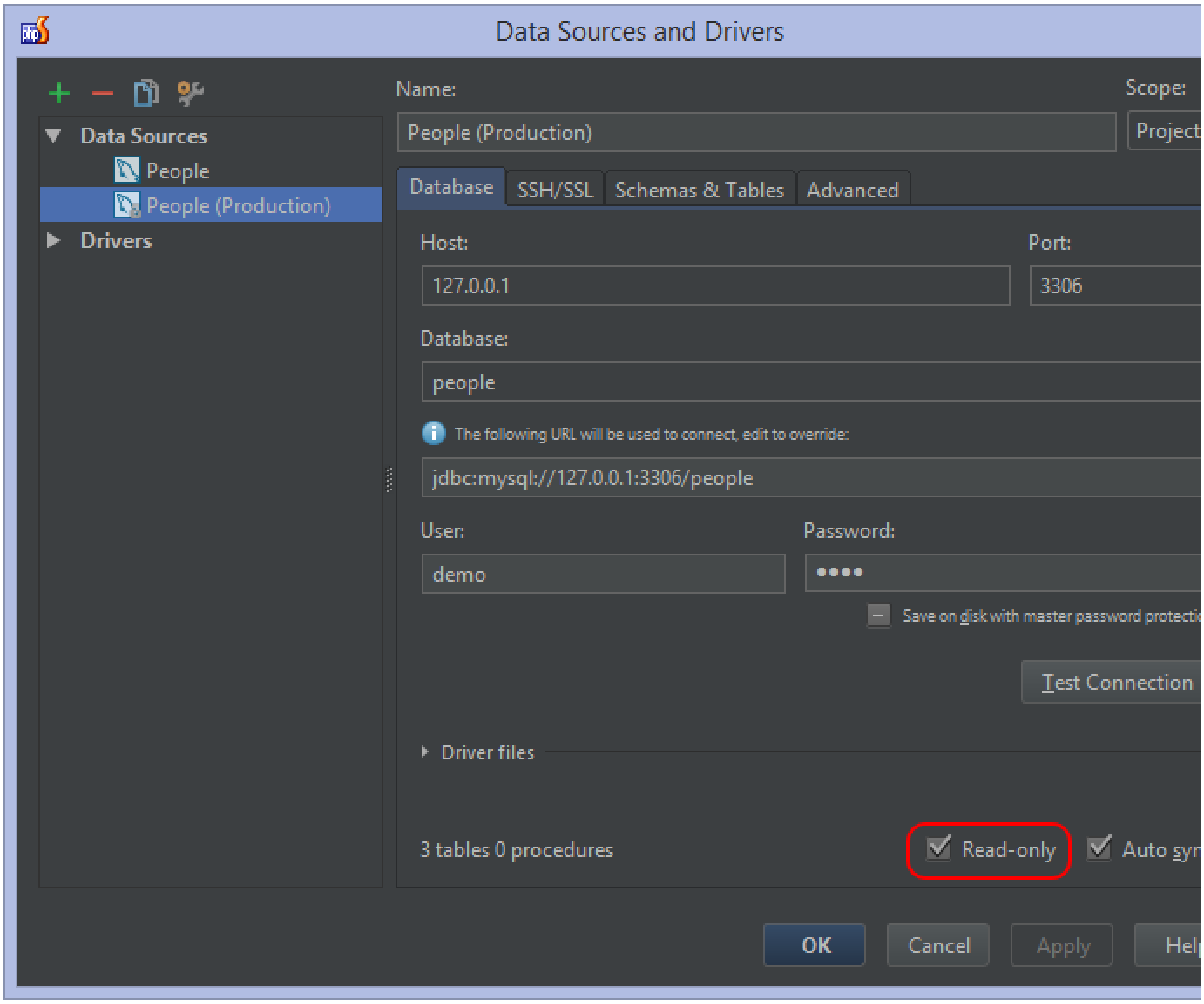
When coloring is enabled, we can clearly distinguish the data source from the tab header by its color. Note that PhpStorm also displays the current database schema in the tab title.



	country_id	name	abbreviation
1	1	Belgium	BE
2	2	The Netherlands	NL
3	3	Spain	ES
4	4	Sweden	SE
5	5	Russia	RU
6	6	Germany	DE

Read-Only Data Sources

When creating or editing Data Source properties, we can mark it as read-only. This will prevent us from accidentally editing or removing data or elements from our schema and only provides read access to the data source, even if the user connecting to the database has write privileges.


Data Sources and Drivers

Data Sources
- People
- People (Production)

Drivers

Name: People (Production)
Scope: Project

Database | SSH/SSL | Schemas & Tables | Advanced

Host: 127.0.0.1
Port: 3306
Database: people

The following URL will be used to connect, edit to override:
jdbc:mysql://127.0.0.1:3306/people

User: demo
Password: ••••

Save on disk with master password protecti

Test Connection

Driver files

3 tables 0 procedures

✓ Read-only
✓ Auto syn

OK | Cancel | Apply | Hel

Tweet