

## Database Setup - Local

@author R.L. Martinez, Ph.D.

### Table of Contents

Overview .....	1
Code Review .....	2
Engine Types.....	4
Inserting Data.....	5
Relation Types .....	5
PKs, FK, and Referential Integrity .....	7
Entity Relationship Diagrams (ERD) .....	7
Creating the falconnight Database in phpMyAdmin .....	9
Creating the falconnight Database Connection in NetBeans.....	11
More Example Databases (from here to document end not on exam) .....	15

### NOTES:

1. The name of the database in this document is “falconnight”. **Substitute** falconnight with the name of your database (e.g. inew2338001001) in all instances.
2. The web-based tool phpMyAdmin is used in the examples below. You are welcome to use other database admin tools such as MySQL Workbench if you prefer. See this link for installing MySQL Server and MySQL Workbench via the [MySQL Installer](#).
3. See this link for MySQL Workbench examples [MySQL Workbench Tutorial](#). There are also many MySQL Workbench videos available via search.

### Overview

We need a database to perform the data-related work in the subsequent tutorials. Begin by creating the falconnight database (use your database name) in phpMyAdmin using the SQL script below. Use NetBeans (or another suitable text

## Database Setup - Local

editor) to code the SQL supplied and save the file as falconnight.sql. We will execute the statements in the file in phpMyAdmin which will create a database with two tables and populate a few rows of sample data.

### Coding, Understanding, & Executing the SQL Database Creation Script

In this tutorial we will accomplish a few things:

1. Create your local database (inew2338001001) from a SQL script file (a.k.a. dump file)
2. Create a database connection in NetBeans
3. More Example Databases

## Code Review

Let's review the code in the SQL script. When studying the SQL code, compare the lines to the ERD (Entity Relationship Diagram) a few pages later. The ERD depicts the relationships between "entities" or tables. Most of the SQL in the script is self-explanatory. For instance, line 1 will create the database falconnight if it does not exist. An alternative to line 1 would be to create the database directly in phpMyAdmin and then begin with line 3 in the code which makes falconnight the current database.

Four tables are created by the script and three records (lines of data) are inserted into each table. Before table creation, the tables are dropped (deleted) if they already exist. Only the first two tables, person and hobby will be required for the tutorial and only lines 1-32 should be coded. Again, for the tutorial, **only code lines 1-32 of the script**. The additional tables can be added later for students who desire to pursue more JSF-JDBC database work.

## Database Setup - Local

```

1 • CREATE DATABASE IF NOT EXISTS `falconnight`;
2
3 • USE `falconnight`;
4
5 • DROP TABLE IF EXISTS `hobby`;
6
7 • CREATE TABLE `hobby` (
8     `HOBBY_ID` bigint(20) NOT NULL AUTO_INCREMENT,
9     `HOBBY_NAME` varchar(45) NOT NULL,
10    `DESCRIPTION` varchar(200) NOT NULL,
11    `CREATED_DATE` datetime NOT NULL,
12    PRIMARY KEY (`HOBBY_ID`)
13 ) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1;
14
15 • INSERT IGNORE INTO `hobby` VALUES (1,'XBox One','Multi-Player, Online, Big Screen',
16    '2014-05-15 00:00:00'),(2,'IPad Games','Portable, Long Battery Life',
17    '2014-05-15 00:00:00'),(3,'Soccer','Excellent Aerobic Workout',
18    '2014-05-15 00:00:00');
19
20 • DROP TABLE IF EXISTS `person`;
21
22 • CREATE TABLE `person` (
23     `PERSON_ID` bigint(20) NOT NULL AUTO_INCREMENT,
24     `NAME` varchar(45) NOT NULL,
25     `NICKNAME` varchar(45) NOT NULL,
26     `CREATED_DATE` datetime NOT NULL,
27     PRIMARY KEY (`PERSON_ID`)
28 ) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8;
29
30 • INSERT IGNORE INTO `person` VALUES (1,'Joe','HandyMan','2014-05-14 00:00:00'),
31    (2,'Bob','Poodle','2014-05-14 00:00:00'),
32    (3,'Bill','Scout','2014-05-14 00:00:00');
33
34 • DROP TABLE IF EXISTS `person_hobby`;
35
36 • CREATE TABLE `person_hobby` (
37     `PERSON_ID` bigint(20) NOT NULL,
38     `HOBBY_ID` bigint(20) NOT NULL,
39     `PERSONAL_COMMENT` varchar(200) NOT NULL,
40     UNIQUE KEY `unique_fk` (`PERSON_ID`,`HOBBY_ID`),
41     KEY `person_hobby_fk_hobby` (`HOBBY_ID`),
42     CONSTRAINT `person_hobby_fk_hobby` FOREIGN KEY (`HOBBY_ID`)
43     REFERENCES `hobby` (`HOBBY_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION,
44     CONSTRAINT `person_hobby_fk_person` FOREIGN KEY (`PERSON_ID`)
45     REFERENCES `person` (`PERSON_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION
46 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
47
48 • INSERT IGNORE INTO `person_hobby` VALUES (2,1,'My favorite game is NBA 2K14'),
49    (3,2,'The IPAD is great for road trips. '), (2,3,'Looking forward to the World Cup. ');
50
51 • DROP TABLE IF EXISTS `personal_history`;
52
53 • CREATE TABLE `personal_history` (
54     `PERSON_ID` bigint(20) NOT NULL,
55     `BIRTHPLACE` varchar(40) NOT NULL,
56     `HIGHSCHOOL` varchar(200),
57     `COLLEGE` varchar(200),
58     PRIMARY KEY (`PERSON_ID`),
59     CONSTRAINT `personal_history_fk_person` FOREIGN KEY (`PERSON_ID`)
60     REFERENCES `person` (`PERSON_ID`) ON DELETE NO ACTION ON UPDATE NO ACTION
61 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
62
63 • INSERT IGNORE INTO `personal_history` VALUES (1,'Austin','Connaly','ACC'),
64    (2,'Dallas','Highline','UT Dallas'),
65    (3,'Marfa','South High','Sol Ross');

```

## Database Setup - Local

The create statements on lines 7 and 22 create the tables. Lines 8-11 and 23-26 create the columns (or fields) in the tables specifying the column name, data type, and special attributes like NOT NULL and AUTO\_INCREMENT. The AUTO\_INCREMENT attribute specifies that the value for that column will be automatically generated by the database and will be incremented with each additional record. The values are therefore unique. For instance, both tables have 3 records. When the next record is inserted in either table, the AUTO\_INCREMENT values will be incremented to 4 for that table. It should be noted that if a record is removed, the AUTO\_INCREMENT values are not recomputed.

On lines 12 and 27 the PRIMARY KEYS (PKs) for each table is specified. It is not required that tables have PKs and their counterparts FOREIGN KEYS (FKs). However, when used, they offer key advantages. First, they provide query performance enhancements for the database engine and are natural choices for indexing which is an essential performance strategy for large databases. Second, PKs are critical for establishing relationships between tables which can provide the protection of referential integrity. More on referential integrity, PKs, and FKs in the following paragraphs.

## Engine Types

The engine type for each table is set to InnoDB. InnoDB is the natural choice when your table routinely requires transaction processing which essentially means that records will be frequently created, changed, and/or deleted from the table. InnoDB tables are ACID compliant and support referential integrity. ACID is an acronym which ensures that transactions will be atomic, consistent, isolated, and durable. ACID compliance is typically required for production database tables that are updated by real-time operations.

On the other hand, if the table is primarily designed for read-only processes, then MyISAM might be the best choice for the engine type. Tables are not required to be locked when the only operations are read actions. However, some degree of locking is required for updates. Another benefit of InnoDB is that it implements row-level locking.

## Database Setup - Local

During updates to an InnoDB table, only the targeted row is locked for the update. MyISAM locks the entire table with create, update, or delete operations. See the sources in the table for more information about these topics.

More Information About Database Topics	
Topic	Source
ACID	<a href="http://en.wikipedia.org/wiki/ACID">http://en.wikipedia.org/wiki/ACID</a>
InnoDB	<a href="http://dev.mysql.com/doc/refman/5.0/en/innodb-storage-engine.html">http://dev.mysql.com/doc/refman/5.0/en/innodb-storage-engine.html</a>
MyISAM	<a href="http://dev.mysql.com/doc/refman/5.0/en/myisam-storage-engine.html">http://dev.mysql.com/doc/refman/5.0/en/myisam-storage-engine.html</a>

## Inserting Data

After each table is created, the INSERT statements on lines 15 and 30 add data to the tables. The IGNORE syntax instructs the query to ignore errors that would be generated if duplicate values were attempted to be added. For instance, suppose the person table already contained the record “2, Sue, Chipmunk”. The record “2, Bob, Poodle” would not be inserted but the system would ignore the error generated and subsequent statements would be permitted to run. If IGNORE were not set, the script would terminate.

## Relation Types

Lines 34-65 are included to demonstrate additional important database concepts. Those lines should NOT be coded for the tutorial. The person\_hobby table is known as a linking table (a.k.a. cross-reference or junction table). A linking table is produced to represent a many-to-many (N:M) relationship.

Let’s first consider the 1:1 relationship between tables, or more precisely, the data in the tables. An example of a 1:1 relationship is that of person:personal\_history. The personal\_history could be contained in the main person table. However, segmenting the personal\_history data into an extension table has the positive effects of potentially improving performance, compartmentalizing data, and preventing or minimizing NULLs in the main table. For instance, both of the HIGH SCHOOL and COLLEGE fields in the personal\_history table could be NULL.

## Database Setup - Local

Relationships and Tables	
Rel.	Description
1:1	person:personal_history. With the 1:1 relationship, exactly one record in each table matches with exactly one record in the other table.
1:M	person:computer. The 1:M relationship consists of one record in one table having many matching records in the related table.
N:M	person:hobby. One person can have many hobbies and a hobby can be enjoyed by many people. Need a linking table like person_hobby to establish the N:M relationship in the database.

The personal\_history table is an example of an extension table which is used with one-to-one (1:1) relationships. An item and its detail table is a standard pattern for 1:1 relationships. For instance, order:order-detail and product:product-detail, are other potential candidates for 1:1 relationships. The names of the tables in those two examples should make the relationship between the tables apparent.

Let's now consider the common one-to-many (1:M) relationship between tables. Suppose we want tables that represent the relationship between a person and his/her computers. The person:computer relationship (which is not included in the falconnight database) would be a connection between the person table and the computer table. Would that be a 1:1, 1:M, or N:M? Naturally, it would be a 1:M relationship since most working adults have a laptop, desktop, smart phone, tablet, or other device that can be classified as a computer.

The last relation type we will consider is the N:M. The 1:1 and 1:M relationships can be established fairly simply between related tables with primary and foreign keys. However, the N:M relationship requires a linking table to connect the two tables to be linked. The one N:M relationship must be represented by two 1:M relationships. Since a person can have many hobbies and a hobby is enjoyed by many people, the example in the falconnight database is N:M. The person\_hobby table is used to link the person table to the hobby table.

## Database Setup - Local

### PKs, FK, and Referential Integrity

When a query is run on multiple tables that are required to return the desired data, an ad hoc join of the tables is performed if the relationship has not been established with primary and foreign keys. However, in the falconnight.sql file, the appropriate primary (PK) and foreign keys (FK) are set so the relationships are established. The PK in a parent table is used to connect to the FK in the child table. The PK, by definition, is unique and NOT NULL.

The PK in the primary table supplies the link to the FK in the child table. The fields are not required to have the same name but data types and content must match. When a relationship has been established between tables using the PK and FK constraints, referential integrity will be enforced by the database. This means that a record in the parent table (the table with the PK) cannot be deleted from the parent table if a matching record in the child table (the table with the FK) exists. Doing so would produce an orphan record (a record in the child table with no related record in the parent table).

The converse condition is another aspect of referential integrity. That is, a record cannot be inserted in the child table unless a related record (with the appropriate PK) exists in the parent table. These checks are supplied automatically by the database engine when a PK/FK relationship is established. Recall that InnoDB tables support transactions and MyISAM tables do not. Therefore, custom code would be required to support referential integrity when using MyISAM tables.

In an educational context, AUTO\_INCREMENT fields are excellent candidates for PKs since they meet the criteria. However, for professional databases, more complex identification fields like employeeID or productID are used for PKs.

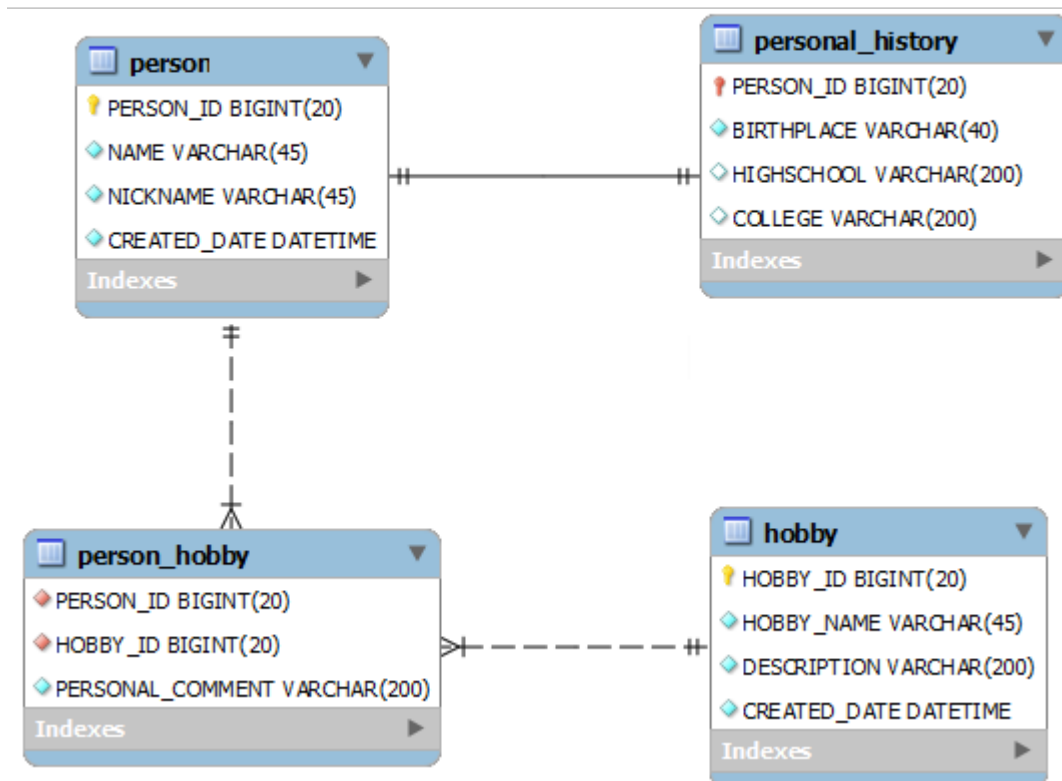
### Entity Relationship Diagrams (ERD)

The ERD is shown and can be very helpful to review and understand the database at a glance. Compare the ERD to the SQL script used to create the database.







## Database Setup - Local

### falconnight Database ERD








The ERD was produced with the MySQL Workbench tool. Workbench has more features and is more powerful than phpMyAdmin. For instance, there is a Designer in phpMyAdmin but it is of limited value compared to the modeling component of Workbench that was used to produce the ERD shown. The additional capabilities come at the price of more complexity which is why phpMyAdmin is used in the tutorials. However, students are encouraged to download and work with MySQL Workbench during or after the semester when their schedule permits.


ERD Symbols Used in Workbench	
Symbol	Description
	PRIMARY KEY
	FOREIGN KEY (may be a Primary Key also)
	UNIQUE
	NULLs not allowed



## Database Setup - Local

	NULLs allowed
	1:1 Relationship
	1:M Relationship
	Identifying relationship – existence of a child row depends on the existence of parent row
	Non-identifying relationship - the PK of the parent must not become the PK of the child.

## Creating the falconnight Database in phpMyAdmin

After coding and saving the falconnight.sql file, open phpMyAdmin and be sure that you are at the “Server” level by clicking the Server: 127.0.0.1 (see highlight). Remember that the local Apache server must be running to use phpMyAdmin. There are two common alternatives to execute SQL scripts. We can open the SQL editor or use the Import utility. For some scripts, the SQL editor can be more reliable. For instance, I have had more success when processing large datasets using the SQL editor than the Import tool. There are a number of settings associated with the Import tool that can be set in configuration files. However, the editor appears to be less restrictive. To use the editor, simply code the SQL directly into the window (or copy and paste) and click the Go button  in the lower right corner when ready to execute the code. Feel free to practice with the SQL editor.

To use the import tool to create the database from the script file click the Import button.



## Database Setup - Local

In the Import window, click the Choose File button and select the file name.


Defaults for the other settings are fine. Click  to execute the SQL in the file.

### Importing into the **current server**

#### File to Import:

File may be compressed (gzip, bzip2, zip) or uncompressed.

A compressed file's name must end in `.[format].[compression]`. Example: `.sql.zip`

Browse your computer:  **falconnight.sql** (Max: 200MiB)

Character set of the file: **utf-8** ▼

#### Format:

**SQL** ▼

#### Format-Specific Options:

SQL compatibility mode: **NONE** ▼

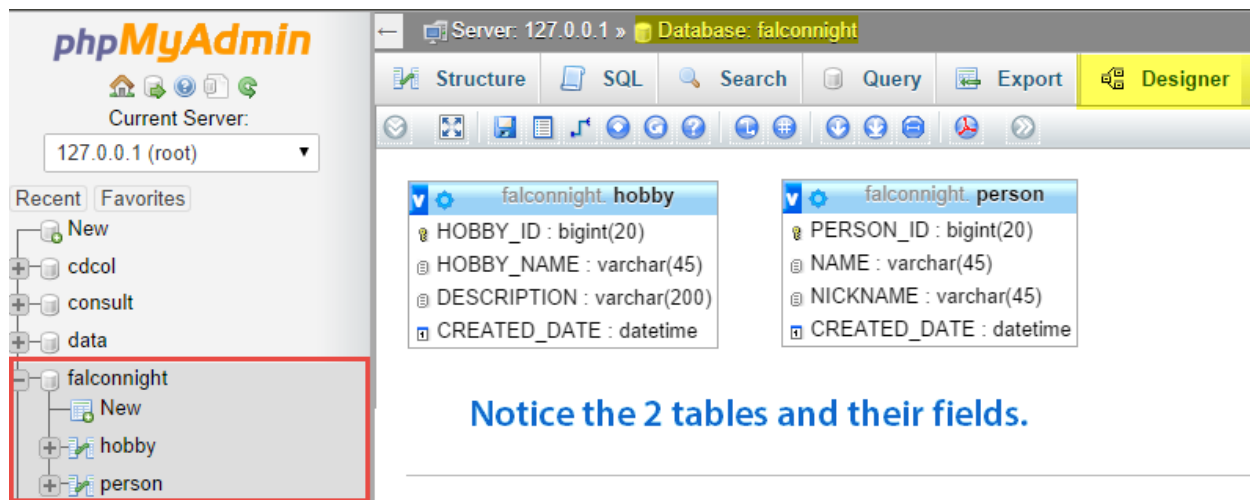
☒ Do not use `AUTO_INCREMENT` for zero values



The falconnight database with two tables is created. Don't be surprised if errors occur due to a mistake you made when creating the file. Carefully examine the error message and review the SQL code. Just like the other code that you have entered and will be entering, every character is critical (except strings, whitespace, etc.). Be prepared to devote the time required to getting your code operating correctly.

Click the falconnight database on the left and it will become the active database as shown by highlight in the bread crumb list. Select Designer to see a model view (ERD) of the database. Review the tables, fields, and data types.

## Database Setup - Local

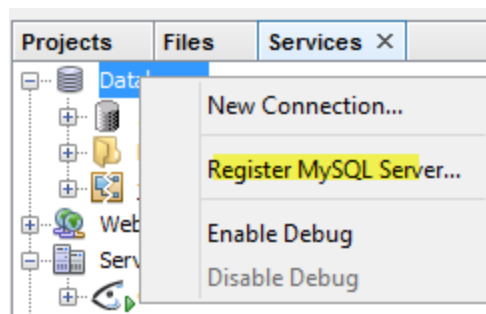


Now that the database is setup in MySQL and we have phpMyAdmin working, we are ready to create the database connection in NetBeans for managing the database via NetBeans.

## Creating the falconnight Database Connection in NetBeans

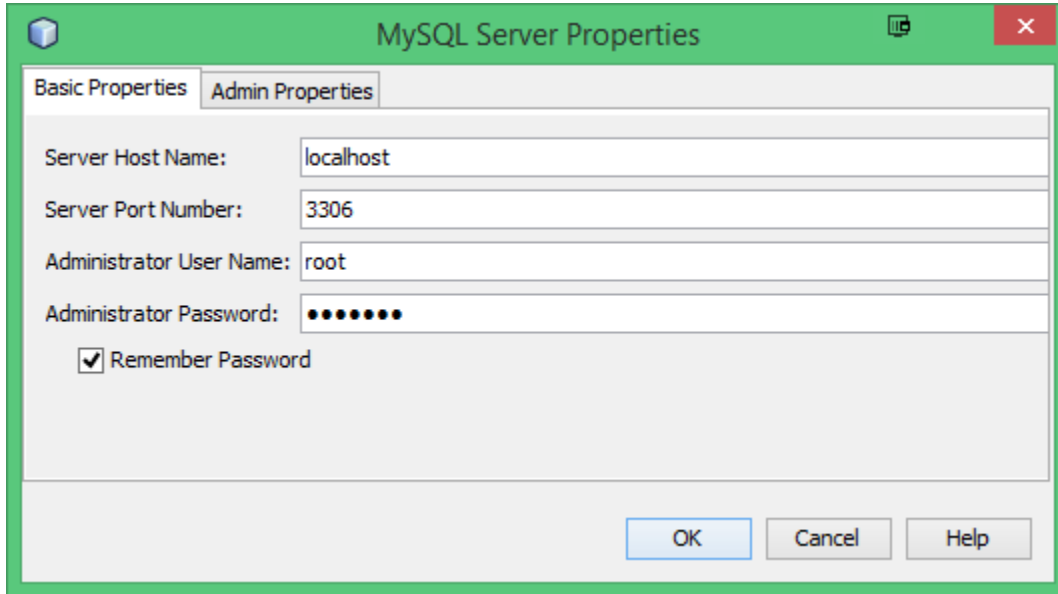
This section is optional. It will not harm the application to setup the connection but it is not required. The important reason that it is not required is that our database connection is maintained by the WildFly sever and “injected” into our application when needed. This concept is called “CDI (Context Dependency Injection)”.

In the NetBeans Services window, R-click the Databases node and select Register MySQL Server...



## Database Setup - Local

Populate the properties as shown. In the dialog shown, the MySQL “root”, or superuser, account is used for the NetBeans database connection. Naturally, you should create another account in MySQL to access the database for more extensive work. The root account is used here to reduce the number of steps.



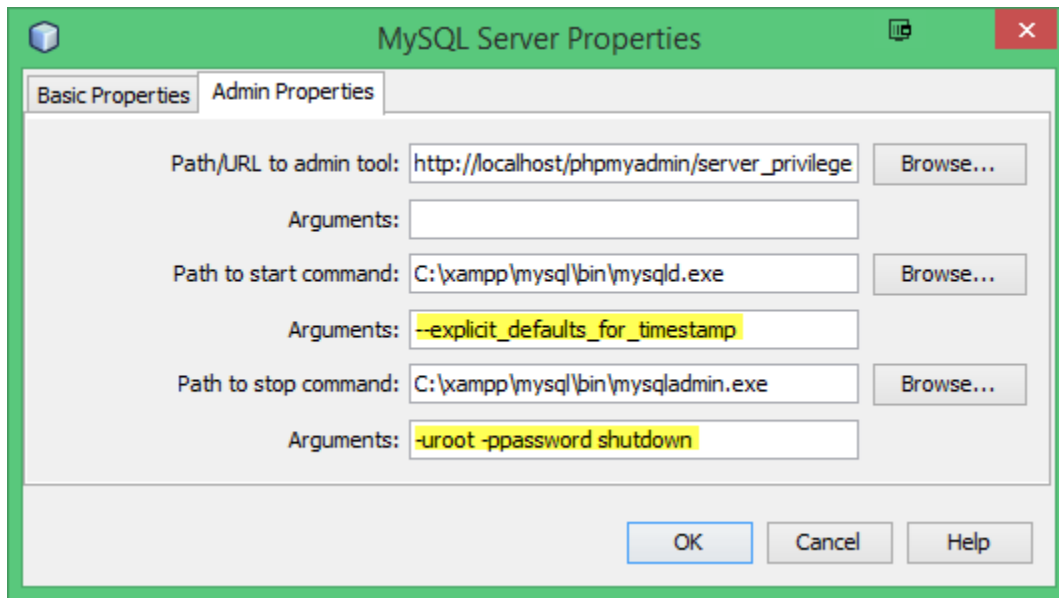
The image shows a screenshot of the 'MySQL Server Properties' dialog box. It has two tabs: 'Basic Properties' and 'Admin Properties'. The 'Basic Properties' tab is selected. The fields are as follows:

Field	Value
Server Host Name:	localhost
Server Port Number:	3306
Administrator User Name:	root
Administrator Password:	••••••••
Remember Password	<input checked="" type="checkbox"/>

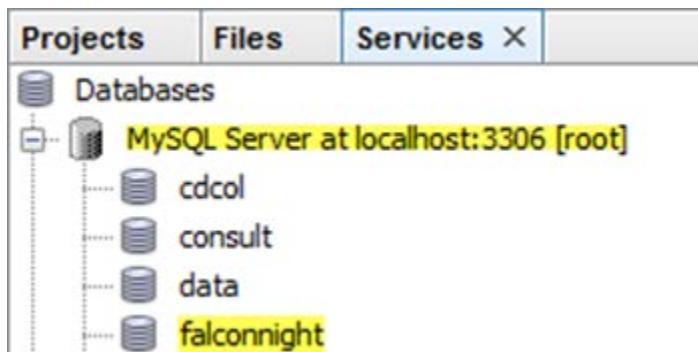
At the bottom right, there are three buttons: 'OK', 'Cancel', and 'Help'.

The Arguments are important for starting and stopping MySQL from within NetBeans. I have better results using the XAMPP Control Panel to administer MySQL (start/stop) but these settings are required to start/stop from NetBeans. Select OK.

## Database Setup - Local

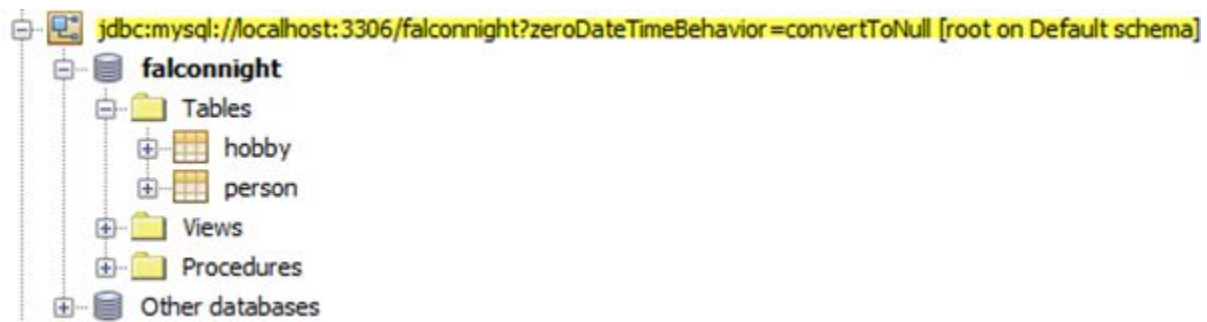


R-click MySQL Server | Connect. Click the plus sign to expand the node and see all databases currently in the MySQL server instance.

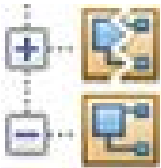


R-click the falconnight database | Connect... to create the database connection and string that will be used by NetBeans to manage the database. However, I prefer using phpMyAdmin (or MySQL Workbench) to manage the database.

## Database Setup - Local



Prior to connecting to the database via NetBeans, confirm the connection is set by the unbroken connection icon (second icon). If the connection is broken, R-click and select Connect... to reconnect.



## Database Setup - Local

### More Example Databases (from here to document end not on exam)

If you would like to work with other databases in the future, samples can be located at the link below. The world database is provided by the MySQL development team at Oracle for practice and training. It is also the database used on the MySQL certification exams and has a reasonably large dataset with which to work.

The sakila database is also a good practice database since it has a number of tables with involved relationships.

<http://dev.mysql.com/doc/index-other.html>

Download the .zip file of the world database

#### Example Databases

Title	Download	HTML Setup	PDF Setup
	DB	Guide	Guide
employee data (large dataset, includes data and test/verification suite)	<a href="#">Launchpad</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
world database (MyISAM version, used in MySQL certifications and training)	<a href="#">Gzip   Zip</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
world database (InnoDB version, used in MySQL certifications and training)	<a href="#">Gzip   Zip</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
sakila database (requires MySQL 5.0 or later)	<a href="#">TGZ   Zip</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
menagerie database	<a href="#">TGZ   Zip</a>		

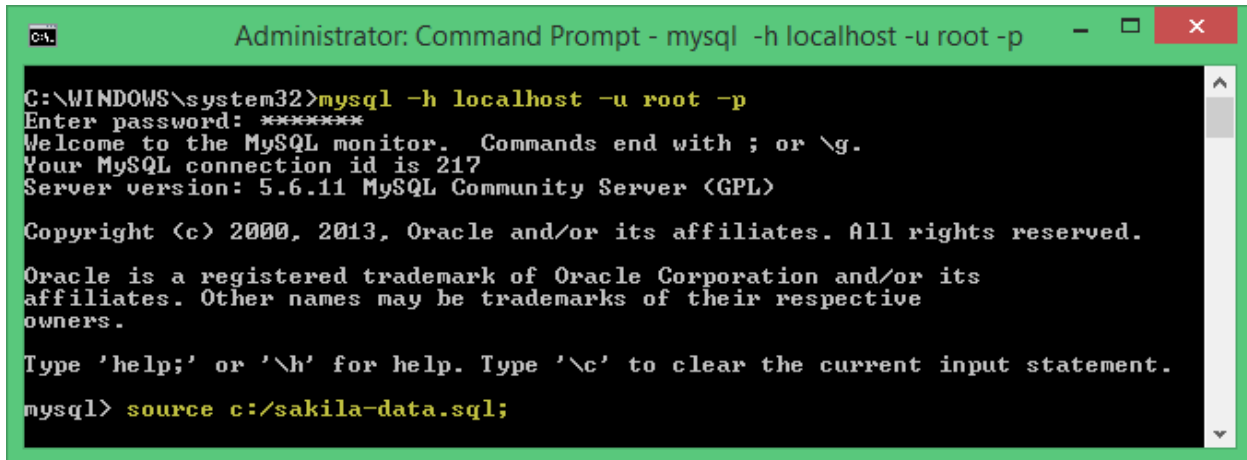
#### Example Databases

Title	Download	HTML Setup	PDF Setup
	DB	Guide	Guide
employee data (large dataset, includes data and test/verification suite)	<a href="#">Launchpad</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
world database (MyISAM version, used in MySQL certifications and training)	<a href="#">Gzip   Zip</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
world database (InnoDB version, used in MySQL certifications and training)	<a href="#">Gzip   Zip</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
sakila database (requires MySQL 5.0 or later)	<a href="#">TGZ   Zip</a>	<a href="#">View</a>	<a href="#">US Ltr   A4</a>
menagerie database	<a href="#">TGZ   Zip</a>		



## Database Setup - Local

Due to the size of the dataset, loading the data for the sakila database in phpMyAdmin by either the Import tool or SQL editor did not work. So, the command line version (MySQL Monitor) was used to load the data. There are configuration settings in the my.ini file which can be modified to allow larger files and other settings.



```
C:\WINDOWS\system32>mysql -h localhost -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 217
Server version: 5.6.11 MySQL Community Server (GPL)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> source c:/sakila-data.sql;
```

@author R.L. Martinez, Ph.D.