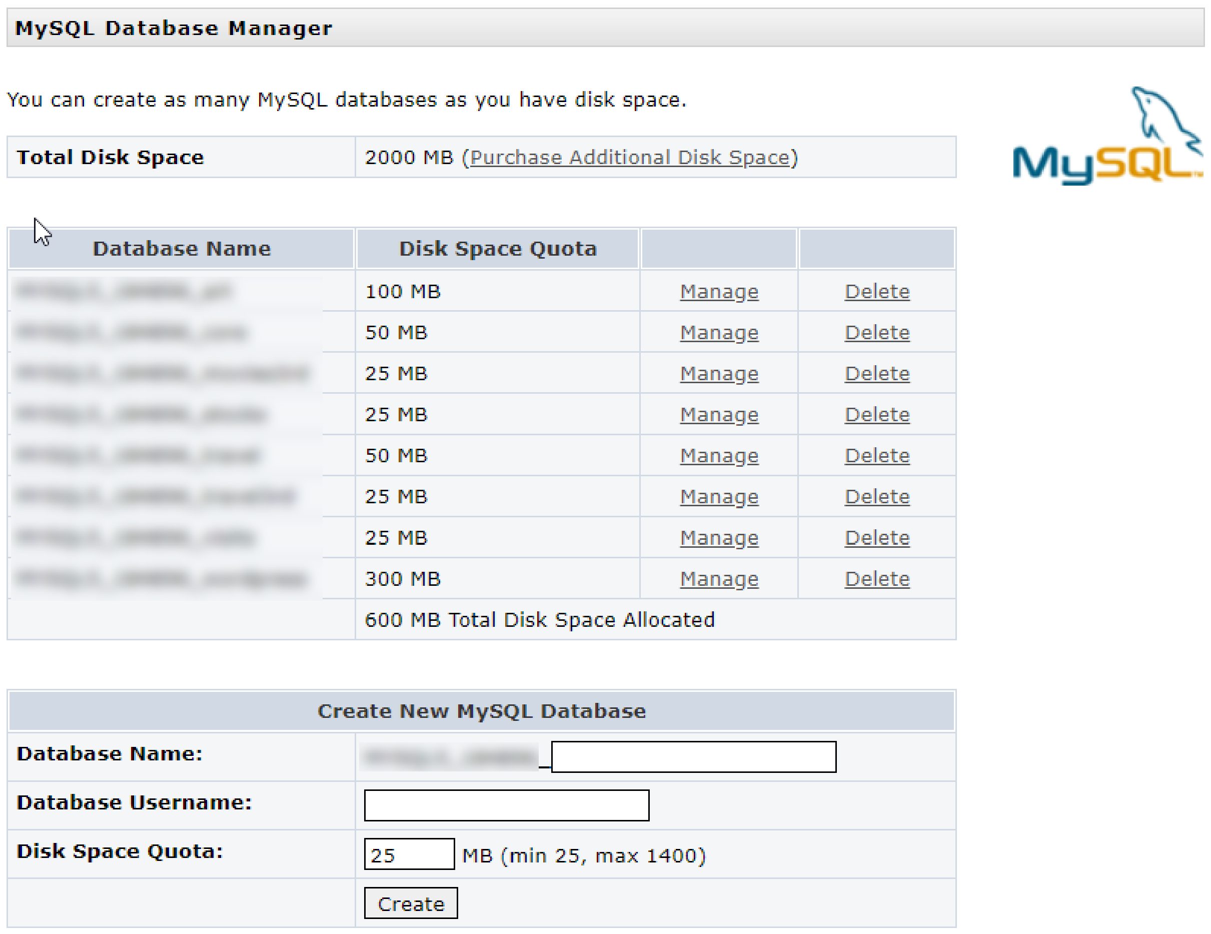
Working with Remote Databases

Connecting to a remote database can be on the complicated side. This document is meant to provide you with some guidance. Every semester I use a different database (art, travel, stocks, or movies) in the course, so the instructions here use the art database; you simply need to replace it with the database being used in your semester.

Creating a Remote MySQL Database

First off, each hosting environment provides its own mechanisms for creating and provisioning your MySQL database. For instance, for my personal web site I have a database manager available through the web host that looks like this.



MySQL Database Manager

You can create as many MySQL databases as you have disk space.

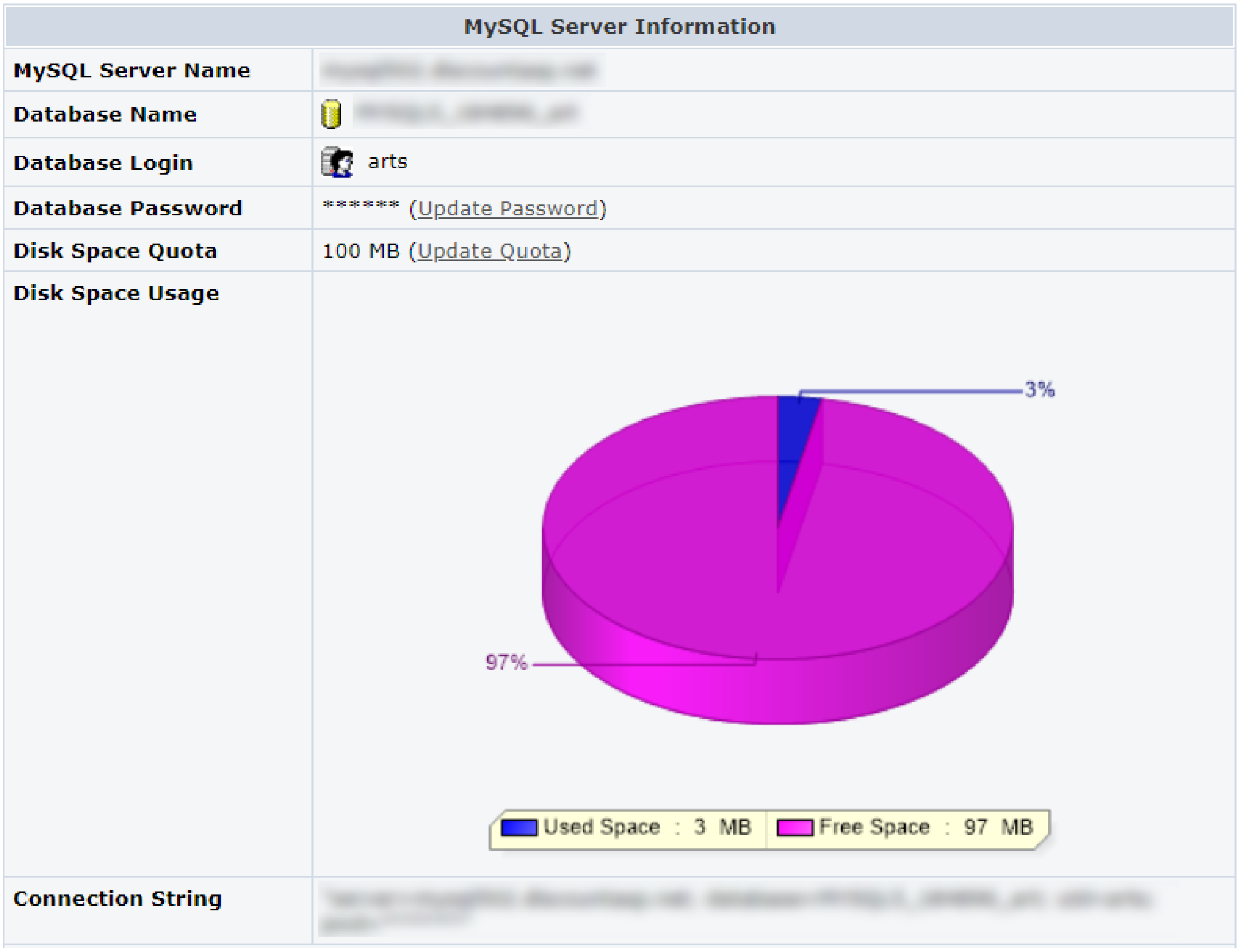
Total Disk Space	2000 MB (Purchase Additional Disk Space)

Database Name	Disk Space Quota		
[redacted]	100 MB	Manage	Delete
[redacted]	50 MB	Manage	Delete
[redacted]	25 MB	Manage	Delete
[redacted]	25 MB	Manage	Delete
[redacted]	50 MB	Manage	Delete
[redacted]	25 MB	Manage	Delete
[redacted]	25 MB	Manage	Delete
[redacted]	300 MB	Manage	Delete
	600 MB Total Disk Space Allocated		

Create New MySQL Database

Database Name:	[redacted]_
Database Username:	
Disk Space Quota:	25 MB (min 25, max 1400)
	Create

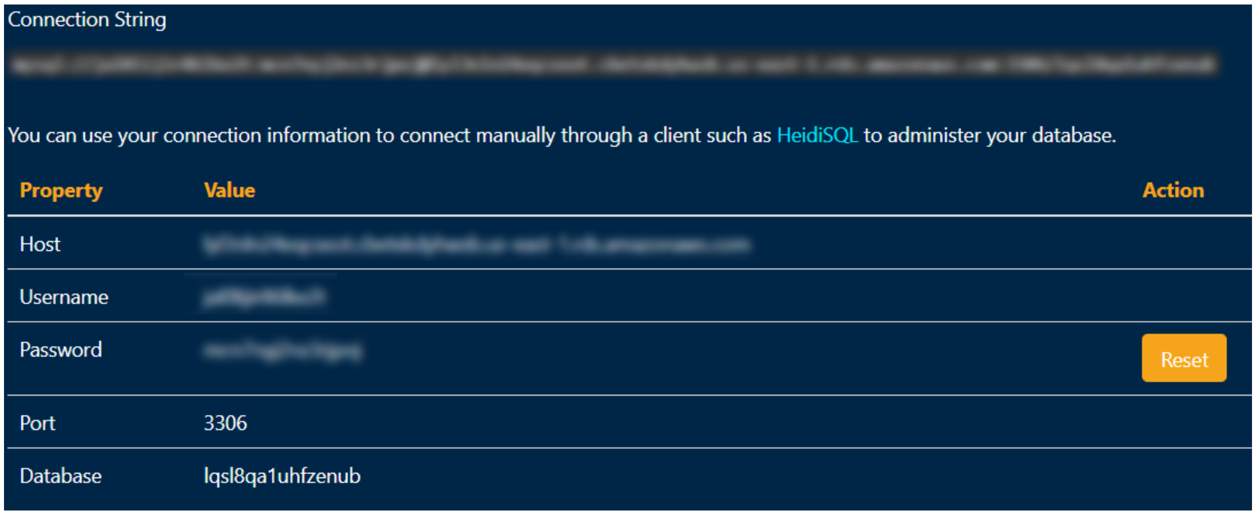
For a given database, I can set the password or see the connection information I might need:



MySQL Server Information	
MySQL Server Name	[redacted]
Database Name	[redacted]
Database Login	arts
Database Password	******* (Update Password)
Disk Space Quota	100 MB (Update Quota)
Disk Space Usage	3% 97% 🟦 Used Space : 3 MB 🟪 Free Space : 97 MB
Connection String	[redacted]

I can then use the database name, login, and password info here when connecting to the database.
Note: I have no control here over the database name or user name. This is going to be important!

I also have a database that I've configured under JAWSDB as part of the Heroku hosting lab that I sent you. I can examine its configuration as well:



Connection String

[redacted]

You can use your connection information to connect manually through a client such as HeidiSQL to administer your database.

Property	Value	Action
Host	[redacted]	
Username	[redacted]	
Password	[redacted]	Reset
Port	3306	
Database	lqsl8qa1uhfzenub	

Note: Once again I have no control here over the database name or user name.

Editing the SQL Import File

Once the database is created, you will need to populate it using the .sql import file. Take a moment to examine the one that has been provided as part of the assignment (e.g., art-3rd.sql, stocks-3rd.sql, etc).

This file contains a whole bunch of data definition SQL statements. Notice in particular the first few lines. They might look similar to the following:

```
DROP DATABASE IF EXISTS `art`;
CREATE DATABASE IF NOT EXISTS `art`;
USE `art`;
```

These lines are **not** going to work for your remote setting because you don't have a database named art in your remote configuration nor will you likely be able to create a database named art. Instead, you have to edit this file to use the names provided for you. For instance, to use my JAWSDB database shown just above, I need to edit my .sql import file as follows:

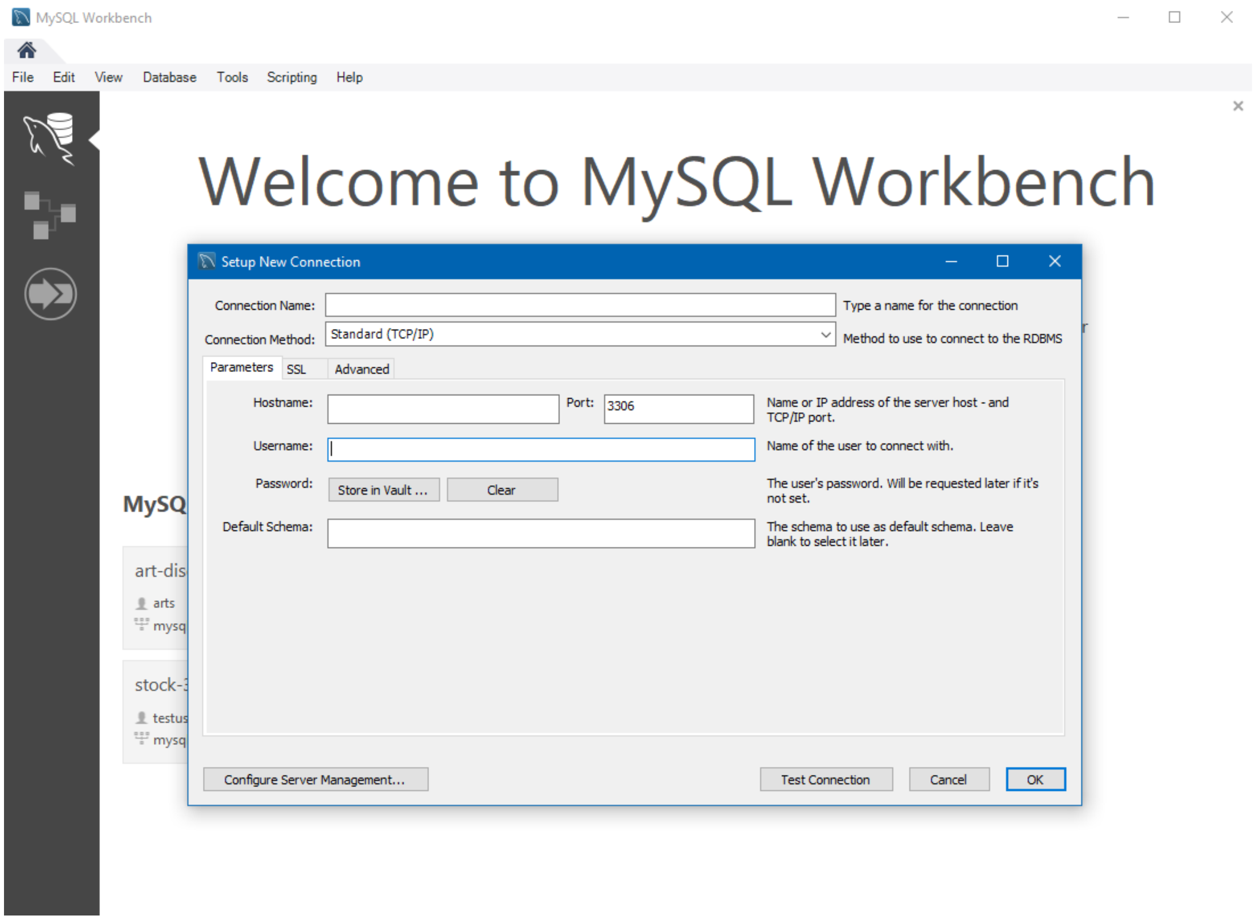
```
# DROP DATABASE IF EXISTS `art`;
# CREATE DATABASE IF NOT EXISTS `art`;
USE ` lqsl8qa1uhfzenub`;
```

Noticed I've commented out the DROP and CREATE statements and edited the USE. You could also simply delete the DROP and CREATE lines.

Populating the Remote MySQL Database

Now that I've modified the SQL import file, I need to "run" it somehow. There are a variety of ways of doing so. You can use the MySQL command line in the /xampp/mysql/bin/ folder. Strangely, this runs fine on one of my computers but doesn't on another.

Instead, I recommend downloading and installing some type of free SQL client. The screen capture above for JAWSDB recommends HeidiSQL. I instead use the free MySQL Workbench from Oracle. When you run it, you can create a new connection (shown below). You simply add in the connection details provided by your host.



Connection Name: Type a name for the connection

Connection Method: Standard (TCP/IP) Method to use to connect to the RDBMS

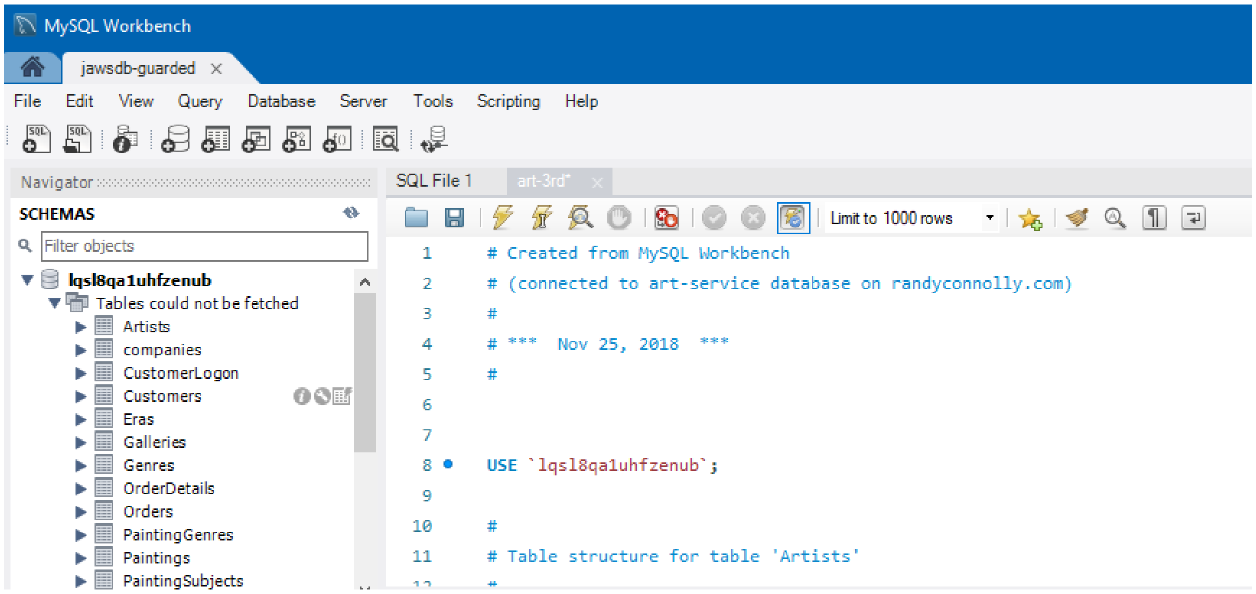
Hostname: Port: 3306 Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

Password: The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

Once connected, you get a dedicated tab where you can open (the folder button) and execute (the lightning bolt icon) the .sql file you edited earlier. Once it is finished, you can see the table structure and run other test SQL statements.



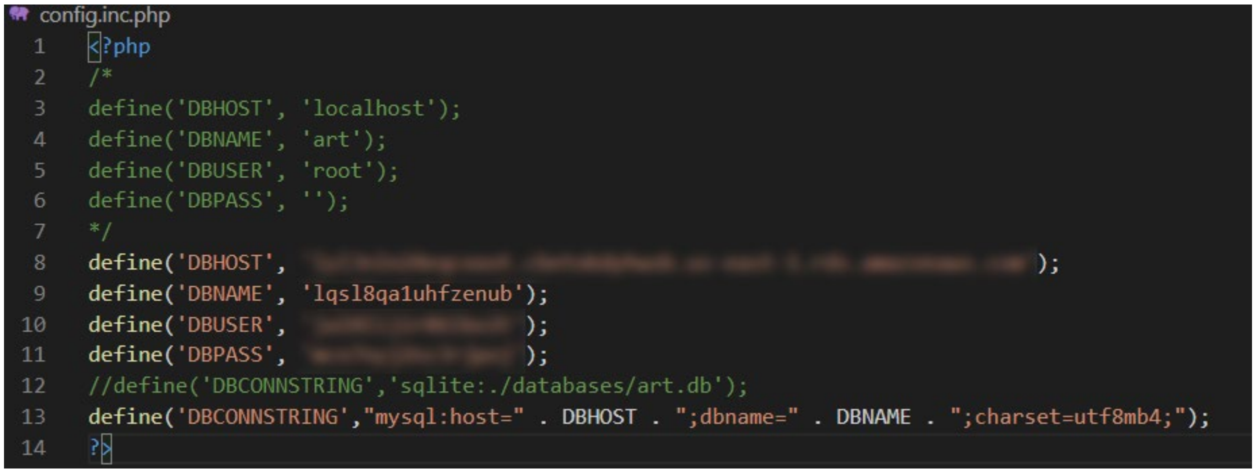
```
# Created from MySQL Workbench
# (connected to art-service database on randyconnolly.com)
#
# ***  Nov 25, 2018  ***
#


USE `lqsl8qa1uhfzenub`;

#
# Table structure for table 'Artists'
```

Connecting your PHP to the Remote MySQL Database

You will need to also modify your connection information in your PHP. I recommend first trying this locally. For instance, I know my Lab14a pages work with the local XAMPP version of the art database. So, I'll begin by modifying the config.inc.php file used in that lab so it references the remote database:



```php
<?php
/*
define('DBHOST', 'localhost');
define('DBNAME', 'art');
define('DBUSER', 'root');
define('DBPASS', '');
*/
define('DBHOST', [redacted]);
define('DBNAME', 'lqsl8qa1uhfzenub');
define('DBUSER', [redacted]);
define('DBPASS', [redacted]);
//define('DBCONNSTRING','sqlite:./databases/art.db');
define('DBCONNSTRING',"mysql:host=" . DBHOST . ";dbname=" . DBNAME . ";charset=utf8mb4;");
?>
```

I then tested one or two of the pages from the lab to make sure they work. If they do, then simply make the same changes to the configuration file used for your assignment and test. Now, when you eventually upload your PHP pages to your eventual host, test again!!! Ideally everything should still work.