

The No Bullshit Bible : Creating Web 2.0 Startups & Programming



**Written by James Gillmore
Edited by Holly Welch**



FaceySpacey

www.faceyspacey.com

MYSQL



TABLE OF CONTENTS

Technical

Chapter 4

MYSQL	2
1. Introduction to databases & SQL	4
2. MYSQL & PHP	6
3. MYSQL & Yii (part 1)	8
4. MYSQL & Yii (part 2)	9
Conclusion & Further Reading	11

4-1 MySQL | INTRODUCTION TO DATABASES & SQL

Ok so SQL stands for “Standards Query Language.” It’s a language to get data from basically tables that resemble Excel spreadsheets. MySQL is a particular application that will exist on your server (like PHP) based on SQL, i.e. that uses the SQL language to query those tables.

100% of the applications that these tutorials are written for will use MySQL. The Yii Framework, which we love, requires an SQL database such as MySQL. All the “models” in Yii mirror mysql database tables. That said, we’re not going to focus on MySQL in terms of its use in Yii right now. We’re going to focus on SQL by itself, and the main things you do with this language.

Let’s dive into some code:

```
SELECT lastname,firstname FROM user WHERE id=3
```

So what could that simple code snippet possibly be doing? It’s selecting a row from the user table where the user id is 3. It’s specifically selecting only the first and last name columns. If we wanted to select all columns we would use the asterisk symbol like this (note: the asterisk in programming often means all):

```
SELECT * FROM user WHERE id=3
```

So that would bring other data in the row, such as the user id itself, email address, phone number, etc.

Now let’s insert a row:

```
INSERT INTO user (firstname, lastname)
VALUES ('James', 'Gillmore')
```

That clearly is inserting my first and last name into their respective columns. SQL is supposed to read like plain english. I'm not even going to explain each part, as it should be obvious. The goal of this tutorial is just to get you familiar with the main goals of SQL, i.e. to select data, insert data, and update data. Let's look at updating now:

```
UPDATE user
SET firstname='Jamie', phone='310-849-8939'
WHERE lastname='Gillmore'
```

That's updating the row with my last name to have my common nick name, and my phone #.

So that should give you a very good idea of what SQL is for. It's all about accessing your data. In reality, in programming the sort of applications these tutorials are for, databases will be the easiest part. Basically you'll find yourself building spreadsheet like tables at the beginning of development of applications, and from then on out for the most part you'll use your framework, i.e. Yii, to query it via PHP code. Of course there will be some times you have to use SQL directly, but by the time you get to that point, you'll be a master and learning more about SQL will be easy. So if a developer you hire makes a big stink about database work he has to do, he's basically stupid. And in reality, he's talking about modeling your product in general, and if you have a good product spec, it should be very obvious how to model your database. A few decisions may need to be made, particularly in terms of preparing for the future, but if you're new to this startup thing, don't worry about the future. Make a good spec and get the product coded up.

So in short, it will boil down to how good your product spec is, and how well your developer understands it. Give your application a user table, a table for its primary data type, such as an Article, perhaps a Category table, add the necessary columns to each table, and bang out your application. It's no biggie because you shouldn't be attempting to code something extremely complex while you're learning and new at this.

To learn more SQL go w3schools (it's where I learned most of the SQL syntax, and I still refer to it regularly): <http://www.w3schools.com/sql/default.asp>

4-2 MySQL | MYSQL & PHP

This tutorial will describe how to get MySQL data into PHP at the most basic level. You will use built-in PHP functions designed specifically for talking to your MySQL application. When I was first learning this, the hard part for me to get was how MySQL and PHP connect to each other. The idea is simply that both are applications on your server, and one application can talk to the other. And PHP specifically made ways to talk to php, i.e. through functions.

```
$username="username";
$password="password";
$databse="your_database";

mysql_connect(localhost,$username,$password);
@mysql_select_db($databse) or die( "Unable to select database");

$query="SELECT * FROM user";

$result=mysql_query($query);
$num=mysql_numrows($result);

mysql_close();

$i=0;

while ($i < $num) {
    $first=mysql_result($result,$i,"first");
    $last=mysql_result($result,$i,"last");
```

```
echo "$first $last";  
$i++;  
}
```

So what this code is doing is:

- 1) logging into your MySQL application via the PHP `mysql_connect()` function.
- 2) then selecting the database to use via the `mysql_select_db()` function
- 3) and then executing a `SELECT` query to grab all users from the database via `mysql_query()` function.

One thing to point out here is that the `SELECT` statement `SELECT * FROM user` is grabbing multiple rows, unlike what you saw in the last tutorial where we used a `WHERE` statement to limit the result to one row. So accordingly we must loop through all the results and echo the first and last name of each user on their own lines.

We use the `mysql_result()` function to access part of the `$result` set via its row # stored in the `$i` variable. The while statement simply loops from 0 upwards until `$i` is not less than `$num`, which via `mysql_numrows()` function equals the total of rows stored in the `$result` resource.

In short, the while statement loops through all the rows in the result set, assigning the first and last names to php variables, and then echoes out the full name to the browser.

The key thing to notice here as you progress to the next tutorial is that this is where the rubber meets the road between your PHP code and your MySQL code. Even if you do end up writing lots of SQL in your Yii applications, you won't have to do all this tedious common coding with functions like `mysql_result()` etc. Yii will automate all that for you, so at most you just have to enter a slick a SQL statement, and at least you don't type any SQL at all.

4-3 MySQL | MYSQL & Yii (PART 1)

Yii allows you to enter raw SQL in various ways. But more importantly Yii also allows you to write Yii PHP code in ways that resemble SQL without having to write any real SQL. First let's cover how to execute SQL statements in Yii:

```
db->createCommand("UPDATE article SET promoted='yes' WHERE id=7")->execute();
```

So here we're creating an SQL command and executing it. The command specifically is setting an article to promoted where the article id # is 7.

We're not going to go too deep into the theory and syntax used here because our goal is simply to point out how Yii automates/simplifies/abstracts what rudimentarily could take many lines of code as you saw in the previous tutorial. The basic idea is that Yii has global tools you can use all starting with "`Yii::app()`". So in this case, we're accessing Yii's database tool with the keyword "db". Then we use it's `createCommand()` method to declare some SQL to use and then the `execute()` method to execute it.

Now let's see how to select data from the database:

```
$rows = Yii::app()->db->createCommand("SELECT * FROM article")->query();
```

Here we simply extracted all the rows from the article table, assigned them to the `$rows` variable (which is now an array containing those rows), and executed the statement in this case with a `query()` method instead of an `execute()` method.

So that's how you can execute raw SQL statements in Yii. It's worth pointing out what's called "fluid syntax" which is being used here. Notice how the `query()` method is attached to `createCommand()` method with "`->`" in between, and even notice how "`->`" is used 2 times before to connect back to the "db" tool and all the way back to

“`Yii::app()`”. What’s happening here is very cool, and one of my favorite things I first got the hang of. Basically each method is a method of an object returned from the previous method. So the `createCommand()` method simply returns a “*command*” object, and it has a `query()` method. You can make these chains on and on if it makes sense.

4-4 MySQL | MYSQL & Yii (PART 2)

Now we’re going to inspect simpler ways of finding and saving data that Yii models have built into them. Yii models fyi are based on the “Active Record” pattern. Look it up some time: http://en.wikipedia.org/wiki/Active_record_pattern . All you need to know for now is the reason that model classes extend from a class called `CActiveRecord` is because Yii uses concepts from the Active Record pattern to design how your models will work.

So anyway the `CActiveRecord` parent class is what gives your models methods like `findByPk()` and `findByAttributes()` that you’re now familiar with. Let’s go back to an example similar to what you’ve seen before in previous tutorials:

```
$articles = Article::model()->findAllByAttributes(array('type'=>'published'));
```

So that code is obviously finder code to return a list of published articles. What’s going on here is that in placement of SQL code such as this “`SELECT * FROM article WHERE type = ‘published’`” the `findAllByAttributes()` method is used. And of course that method takes as a parameter an array with data about what to search for. The idea is that this is native PHP coding, and no SQL is required. It will make coding a lot easier and save you a lot of time.

Now let’s get into saving newly created articles:

```
$article = new Article;
```

```
$article->title = 'The FaceySpacey Programming Bible';  
$article->pageCount = 300;  
$article->save();
```

See how natural that last line is. You can simply create an object, and call its `save()` method, rather than write this SQL to save it:

```
INSERT INTO article (title, pagecount)  
VALUES ('The FaceySpacey Programming Bible', 300)
```

The Active Record tools of Yii allow you to think in terms of PHP code, and more specifically in terms of Yii code, rather than Yii code, PHP mysql functions, and Mysql. It allows you to not have to jump back and forth between different languages, and think in terms of just a few core method and the parameters you pass to them. Basically it means there's a lot less to remember, and what you do have to remember is the most powerful stuff that offers the most flexibility. In reality, in terms of finder methods such as `asfindAllByAttributes()`, it will all be about the parameters you pass to it. And these parameters are usually the application-specific pieces of information that are easy to remember such as table column names since it's your application, rather than tons of coding syntax.

Conclusion & Further Reading

We at FaceySpacey hope you've enjoyed our FaceySpacey Bible, and are coming away many times more ready to succeed at your next software startup. At the very least, you should have a birds-eye-view of what you need to do to get your startup, and have quelled a lot of insecurities you may have had regarding how you should execute it. That said, I will point to you to what you should do next.

As promised, the following is a list of the precise books I read to master web development using HTML/CSS, Javascript, PHP & MySQL. They are presented in the best order to most efficiently learn the subject at hand. It's similar to the order I read them in, but enhanced based on what I learned and the order I wish I read them in.

Good luck:

HTML/CSS:

CSS Mastery: Advanced Web Standards Solutions

<http://www.amazon.com/CSS-Mastery-Advanced-Standards-Solutions/dp/1430223979/>

Before you start coding PHP, Javascript, etc, understand how HTML works. This is where you start. HTML is easy. Read this book in combination with studying the HTML & CSS tutorials on w3schools.

PHP & MySQL:

PHP & MySQL For Dummies, 4th Edition

<http://www.amazon.com/PHP-MySQL-Dummies-Janet-Valade/dp/0470527587>

This book--well an older edition--I read a year before I got serious about learning to code. I read it and didn't actually code anything i learned, but what it did was plant seeds in my head with regards to what programming is all about and what databases are all about, and how to connect the two. It assumes very little in what you may

already know, and is an excellent start in your journey to becoming a master programmer.

PHP Object-Oriented Solutions

<http://www.amazon.com/PHP-Object-Oriented-Solutions-David-Powers/dp/1430210117>

This book is where I learned what OOP is. I didn't get the hang of it until reading the following book. Don't worry if you read this and have a hard time with it. This book and the next each have introductory chapters that go over how OOP works. It took me reading basically this book and the next book about the same stuff to get it. This book is a lot less complicated than the following and dives into practical examples & problems, whereas the next is a lot more theoretical.

PHP Objects, Patterns and Practice

<http://www.amazon.com/Objects-Patterns-Practice-Experts-Source/dp/143022925X>

After reading this book, I basically mastered OOP. It's a very hard book to get through if you're new to OOP, and goes into some very advanced stuff, specifically tons and tons of "design patterns." The design patterns are presented in as basic of a form as possible, but they weren't very practical like examples from the previous book in that you probably will never actually need any of the code used in the book. Either way, this is my favorite Programming of all time because it taught me how to think like a coder and how to solve complex problems with concise refactored solutions. It really showed me what is possible with OOP. You don't know PHP unless you've read this book.

Pro PHP: Patterns, Frameworks, Testing and More

<http://www.amazon.com/Pro-PHP-Patterns-Frameworks-Testing/dp/1590598199>

I read this book too just to solidify my experience with PHP, and cover all my bases. Check it out. It's optional.

PHP Functions Essential Reference

<http://www.amazon.com/Functions-Essential-Reference-Torben-Wilson/dp/073570970X>

At some point during my study of PHP I found this book and decided just to learn every PHP function available so that I could better understand the examples in the above books. Start reading this early on, and complete the whole thing. You'll quickly learn patterns in how PHP functions are named, and as a result be able to guess what a function does within the context of the examples in the above books--even if you don't remember precisely what it does.

Agile Web Application Development with Yii 1.1 and PHP5

<http://www.amazon.com/Agile-Web-Application-Development-PHP5/dp/1847199585>

I read this book in combination with reading the Blog Tutorial and Definitive Guide on YiiFramework.com. When you're done studying all these materials, you'll be amazed with how much power you have. This book isn't hard to read either. You'll love it if you reach this stage!

Javascript & jQuery:

Learning jQuery, Third Edition

<http://www.amazon.com/Learning-jQuery-Third-Jonathan-Chaffer/dp/1849516545>

jQuery is a framework built on top of the native browser language of Javascript. Usually one would recommend you learn the base language--Javascript--before learning an abstracted framework on top of it--jQuery. However because of the nature of jQuery and how comprehensive it is and because of how quirky Javascript is coming from PHP, I found it best to jump to jQuery and immediately start accomplishing the DOM manipulation tasks I needed. And ultimately because of syntax similarities between PHP and Javascript I was able to get productive in Javascript without studying a single book just on Javascript. One thing I did different

when studying this book from the PHP books is I did every single tutorial as I read it. The reason is because when I learned PHP, I was learning my first real programming language--so it took me a lot of time to just digest things before I could code a single line, which is why I just read PHP book after PHP book before I got started until it all made sense. However, by the time I got to Javascript & jQuery, I understood how programming in general works and found it helpful for memorization purposes to immediately start doing the tutorials.

jQuery 1.3 with PHP

<http://www.amazon.com/jquery-1-3-PHP-Kae-Verens/dp/1847196985>

With this book I didn't do all the tutorials like I did with *Learning jQuery*, but what reading this book did for me is taught me precisely how Ajax works and what it's all about. After reading it, coding features that required Ajax using Yii and PHP was obvious and a no-brainer.

JavaScript: The Good Parts

<http://www.amazon.com/JavaScript-Good-Parts-Douglas-Crockford/dp/0596517742>

This book gave me a deep understanding of the Javascript language and what it's truly all about. After reading it, many hours of debugging and head-scratching when coding Javascript & jQuery were removed from my schedule--because I finally learned the quirks of the Javascript language I needed to know.

Pro JavaScript Design Patterns

<http://www.amazon.com/JavaScript-Design-Patterns-Recipes-Problem-Solution/dp/159059908X>

Now this book took my Javascript game to the next level, gave me an idea of how jQuery was built, taught me how to do things similar to how you would in a "classical" OOP language like PHP, and completely ended any remaining head-scratching I was having with Javascript, particularly with how "scope" works in Javascript.

Linux:

The Official Ubuntu Server Book, 2nd Edition

<http://www.amazon.com/Official-Ubuntu-Server-Book-2nd/dp/0137081332>

Note: by the time I read this book I had already learned Linux through blogs on the internet. The best thing I can recommend you do is install Linux on your computer from the Ubuntu website, and start navigating around the command line, practicing Linux commands you learn off the web. Just google "linux tutorials" and you'll be off to a running start. That said, by the time I got proficient in Linux and after I read this book, I felt confident that I really knew what I was doing and had practical solutions for the most common problems you'll face at the command line.

Apache Cookbook: Solutions and Examples for Apache Administrators

<http://www.amazon.com/Apache-Cookbook-Solutions-Examples-Administrators/dp/0596529945>

This book I treat like a pocket reference and still refer to it often since it's impossible to remember all the different Apache configurations, given how comprehensive this web server application is. I did read it through when I first got it. I kinda skimmed it though-- just to get an idea of what is possible. Getting an idea of what is possible without mastering a subject matter is so important in programming because you'll know where to look when you face a challenge that the subject matter can solve.

Pro Bash Programming

<http://www.amazon.com/Bash-Programming-Experts-Voice-Linux/dp/1430219971>

This is a little advanced for readers of the *FaceySpacey Bible*, but I'm putting it here because it really took my Linux skills to the next level.

NON-TECHNICAL BOOKS:

Smart and Gets Things Done: Joel Spolsky's Concise Guide to Finding the Best Technical Talent

<http://www.amazon.com/Smart-Gets-Things-Done-Technical/dp/1590598385>

If you plan to grow a small application into a large company, this book is a must. It's short. Read it.

SEO Book.com

<http://www.seobook.com>

This obviously isn't a book, but I read their entire site like a book, and its creator, Aaron Wall, expects you to read it like a book. When I was done reading it, I felt I was completely up to speed regarding what SEO is, how search engines work, and practical techniques to get better rankings in search engines.

For daily Startup Wisdom, checkout [FaceySpacey.com/blog](http://www.faceyspacey.com/blog) daily. And don't forget to download the entire No Bullshit FaceySpacey Bible or more individual chapters here:

<http://www.faceyspacey.com/resources?section=book> .

Á
V@) | • Á* aq Á/ { Á&^ ù] æ^ Áq aÁ^ Á~ /^ Á[& @ & Á ~ o [FaceySpacey.com](http://www.faceyspacey.com) Á -e} Á
{ /Á' /c@ /Á} [, /á* ^Á ^Á & \ Á[Áa ^Á [^ /Á ùæc'] Á[Á @ Áæ • Ñ
Á