Training Application Developer

Learning Trajectory Database

Connect to the database

Domain B Level 2

Author: Benn Bergmann, Erik Mast

Date:31-5-2023

Table of contents

Overview

Level: Domain B Level 2
Duration:2 weeksMethod: Weekly schedule

Prior knowledge

Domain B Level 1 Introduction Database

Domain A Level 1 Programming PHP

Combined study

Domain A Level 2

Materials

- Your laptop with;
- Editor, for example Notepad, Notepad☺++, Atom (: recommended!)
- Web browser, e.g. Internet Explorer, Firefox, Chrome
- XAMPP

Sources

- See Appendix Sources. Before you start, at least read the sources.

Instructions

- Create SQL commands
- Search sql credentials
- If the text says that you have to read through something from the sources, you should actually do so.

Description

In this command, you'll learn how to use SQL to retrieve and store data in the database. First you learn the different SQL statements and then you use them when creating the final assignment.

You will also learn how to add rules to a table, discard rules and customize rules, and make an overview of all the rules from a table.

Goals

After this period you will be able to make simple SQL statements and apply them to the database. You will also be able to make contact with a database and extract and store data from it.

Assessment

This assignment is assessed on the basis of the final assignment and a conversation about the end product.

Study components

SQL statements (+/- 32 hours)

SELECT, INSERT and DELETE statements.

Theory

The concept of CRUD is central to the use of databases. This is an abbreviation of

"Create, Read, Update and Delete"

Translated, these are the basic actions you can do on a database, namely:

What does it do	Abbreviation	SQL command
Create a new rule in a table.	CREATE	INSERT
Read one or more lines from a table.	READ	SELECT
Customize a rule in the table	UPDATE	UPDATE
Remove a rule from the table	DELETE	DELETE

Each of these SQL Statements has its own syntax (how to correctly issue the command to a database). This syntax can be found in the sources.

Introduction to practice assignments

In the assignment below, you are always asked to save the SQL statements you have made in a text file. Do that too, because it will be asked for.
You are always asked to type out the assignments completely. This is necessary to get practice in creating SQL statements. If you use copy and paste, that will become a problem later, because you have to look up how it works every time. And because of that, you may be fooled by your colleagues.

What do you call a table?

There is a discussion going on on the internet between technicians about what you should call your tables.

1. Plural (e.g. songs)
    a. because there are more songs in a table.
    b. It reads better in a query
2. Singular (e.g. song)
    a. Because the table describes a relationship and not a thing (might be vague at the moment)
    b. It reads better in the "where" of the SQL command
    c. Each record is a representation of one thing.

Whatever you think, always make sure your naming is consistent. And be flexible withthe nature of your team, being consistent is more important than whether one person is right

Practice assignment 1

Create a new database in PhpMyAdmin and name it "db_level2_opdr1". Create a table ("songs") in this database in which you can store your favorite songs.

This table has a column with a unique Id field (by now you know how to do that), a field with artist names("artist"), and a field with song names("title"). Determine the data type of the fields yourself.

a. Fill this table with at least 5 songs from at least 2 different artists. Of course, more is also allowed. Do this by using the INSERT command, and type the command completely each time.
   This is important because you need to practice this. Tip: Create a separate text file and save the SQL command you executedin it so you can use it again later.
b. With each line, change the name of the artist or the name of the song. Also do this at least 5 times with the UPDATE command, also type this command completely each time. Also, save these commands in the text file. After each command in PhpMyAdmin, check whether the desired result has been achieved.
c. Use the SELECT command to create an overview of all songs, also save this command in your text file.
d. Use SELECT to see all songs by one artist. Also save this SQL in your text file.
e. Use SELECT to see one song. There are more possibilities for that, try to think of which one and try it out. Keep these in your text file as well.
f. What's the most convenient way to show one song. Save this query and add why.
g. Use DELETE to delete two favorite songs. Keep these in your text file as well.
h. Export the database and save the result at the bottom of your text file.
i. Check your export by importing it into a **new** database.

Connect to the database (+/- 16 hours)

If you have XAMPP (or another "personal" web server such as USBWebserver) installed, you have a database server in your computer and a web server. If you combine them, you can obviously do much more. It makes sense that this can be done, now let's deal with how to do something like that.

Theory

Before we can really make contact with the database, we have to talk about **Objects**. We'll go deeper into that later, but now we'll take a quick look at it:

Objects in brief

An object is actually a programming unit. The most important feature is that it bundles code and properties. You have to think of it a bit as a variable (like an integer oid) that can manage its own affairs. Just as you can come to school independently, you know how to get there, which bus lines you may have to take, and which way those bus lines go. When we say "Go home" to you, you know where your house is, and how to get there.

Connection object

If you want to connect to the database, use a connection object. Before you have to connect, you will first have to find out some parameters (data) from your own database server. We will explain step-by-step how it works here.

First of all, we create a regular php file

```
<?php
```

You use four variables, one for the server (localhost which is your own computer), one for the name of the database, one for the username and one for the password. This is just like with all kinds of other systems, such as the school portal.

Of course, you have to fill these variables with the values that apply to your computer.

```
$servername = "localhost";
$databasename = "database";
$username = "username";$password = "password";
```

Explanation of connection parameters MySQL.

Before you can establish a connection to the database, you will first have to give the four variables from the code the correct values.

$servername
This is the name of the server where the database is located. In the case of an XAMPP hosted server, the correct value here is "localhost", or the server running locally (on your laptop).

$databasename
The name of the database you want to connect to. If, for example, you want to use the database from the practice assignments, the correct value here is "db_level2_opdr1".

$username & $password
To make a connection, you also need a user account that has access to the database.

Accounts
On the phpmyadmin homepage you will find a link "User accounts" at the top. Here you can find information about the existing user accounts or you can create a new account. Make sure the account you're using has a username, password, and permissions to connect to the database. Connecting to the root account is **not desirable** and a bad habit. The root account is a special account that has all the rights. So it is also a paradise for **Hackers**. You can easily create an account and create a corresponding database.

Note: The "%" character in "server name" field means that the account has full access through all servers except localhost. This can cause problems when connecting to your own laptop (localhost.

Then you create a connection object (see below). What happens here is that a piece of code with properties is stored in the variable. The database is already being contacted in the background. The nice thing is that you can then give the variable commands. But more on that later.

```
Create connection
$conn = new mysqli($servername, $username, $password, $databasename);
```

Because something apparently happened in the background, you can then immediately ask if it went wrong. And if something did indeed go wrong, you can ask php to stop immediately and report the error. And if nothing goes wrong you can let php continue.

```
Check connection
if ($conn->connect_error) {
 die("Connection failed: " . $conn->connect_error);}
 echo "Connected successfully";

?>
```

Practice assignment 2

Use the code in the theory above, and try to get it working for your laptop. Use the database "databaselevel2opdr1" that you created in assignment 1.

Theory

The object you created by contacting the database can be given all kinds of commands:

$conn->query($sql)

This command executes a SQL command on the database server. The result is also an **object** that you can also give orders to!

$conn->close()

This command is to shut down the connection, when you're done. That's neat.

There are many more assignments, which can be found in the sources.

The following item

The result of $**conn->query($sql)** is also an object as mentioned above.

You can also give that object all kinds of interesting commands such as **$result->fetch_assoc()**. And the outcome of that is an **associative array**. An array in which you can find the different values based on a name and not an index, as is the case with simple arrays.

Practice assignment 3

Show off your favorite songs

Read "MySQLi for beginners" from the sources carefully to "Close that connection", and create a web page, which shows the content of the "favoritesongs" table. Formatting is not important, but if you want to practice extra with <TABLE> tags, go ahead and do so. It will make your teachers very happy.

Also decorate this file with some formatting in CSS, and make it a neat HTML file.

Theory

You've had some practice with creating **INSERT** queries before. The last thing the user wants is to type in these types of queries and hope that it ends up in the database.

You can run an INSERT query in the same way as the summary query, but of course no result will return.

It is more convenient that you have an input form for this. Take a look back at the Programming Level 2 module how you created an HTML form there. You will combine this knowledge with the knowledge above.

Practice assignment 4

Create a new php file, in which you can enter a favorite song, and make sure that it is processed in the database. If the processing is successful, jump back to the file where you can see the overview, which is the file you created in the previous practice assignment. Also, make sure you have a button in that file that jumps to your add form. For the form, you also format it with CSS and that it fits your previous screen.

Practice assignment 5

Make a copy of the form from assignment 4 (just in case)

On the overview screen, place a button for each song, with which you could change that song. See if you can adjust the form from assignment 4 so that the details of your song are filled in that form. For example, you can use a session variable or parameters in the address line (you have to know both techniques eventually, so explore both possibilities) or in a different way. Answer the question: **How could that be?**

Practice assignment 6

You can add data and create an overview, and you can call a screen with the data already filled in. Now adjust this in such a way that the changed data from the form also ends up in the database. Hint: you have to use an **UPDATE** query, go find out how that works if you haven't already. You have to program that query and the changes you make to the database with the form must be visible in the overview screen.

Practice assignment 7

Place another button next to each button in the overview screen. Make sure that button always discards the selected song and that you then return to the overview screen.

Summarized

Now you've seen all the elements of a CRUD once:

C: Adding a rule

R: reading one or more lines

U: Changing a rule

D: Deleting a rule.

Now it's time to practice some more

Practice assignment 8

Now create a total set of files for CRUD with a table where you can keep track of your birthday wish list. Make sure you have a field for price, description, and where it's for sale, and a field for a web address for that store. Think carefully about what kind of fields you will use.

Final assignment (+/- 16 hours)

Create a CRUD set for the birthdays of your friends and family. Make sure you record the date of birth and also that you calculate how old someone will be. You will need fields for first and last name and date of birth. The challenge here is obviously the age, you will be in awe if you can also print the current age in days and / or months in the overview screen, but beware! That's not something you keep in your database.

You hand in the assignments (both practice assignments and final assignment) in Magister. Always make sure the website is sent packed (zipfile) and don't forget to export and send the SQL. If you don't know how to create a zip file: Right-click on the folder where your files are located, and then choose "Copy to"->Compressed (zipped) folder. A zip file will then be created that you can send.

Sources

.SQL

- ✓ **Create, Read, Update and Delete**
  https://en.wikipedia.org/wiki/Create,_read,_update_and_delete

- ✓ **SQL Tutorial**
  Read this up to and including the chapter "SQL Delete"

  https://www.w3schools.com/sql/

- ✓ **PHP: MySQL Database**
  https://www.w3schools.com/PhP/php_mysql_intro.asp

Connect

- ✓ **PHP Connect to MySQL**
  https://www.w3schools.com/php/php_mysql_connect.asp

MySQLi reference

- ✓ **PHP5 MySQLi functions**
  https://www.w3schools.com/php/php_ref_mysqli.asp
- ✓ **MySQLi for Beginners**
  https://michael.wright.uk/php-mysqli/

Andkelve or plural in the table name

- ✓ **Use singular names nouns for table names**
  https://www.teamten.com/lawrence/programming/use-singular-nouns-for-database-table-names.html