# Homework: MySQL and PHP Fundamentals

This document defines the homework assignments for ["PHP Web Development Basic" Course @ Software University](https://softuni.bg/trainings/1746/php-web-developmentbasics-september-2017). Please submit ……………………………………………….???

# Part I. Related Tables

In this section you will solve problems by which you will exercise **table relations** (Primary and Foreign Keys). Every problem is a **task for an HR Application**. You will need the **geographic\_database.sql** to work with. Download it from the web page and section for this exercise:

**10. PHP-MySQL-and-PHP-Fundamentals-Exercise-Databases.zip**

You can load **geographic\_database.sql** into the server as shown on the screenshot.

# Problem 1. Database for HR Application

Janette works as a HR (Human Resource) in a company called BestWishComeTrue.com with about 5000 employees from people all over the world. She needs to keep track all of the employees. Write a simple application which can save in a database three different things:

1. Employee **id**, **First Name**, **Middle Name**, **Last name**, **department**, **position**, **passport ID** (the number on the employee card issued by the state). The employee can work only at one department and have only one position until he/she is moved to another department and position. The combination of names and passport ID is unique. Janette uses the simple **id** (**primary key** and **auto increment**) not Passport ID.
2. Employee email addresses. It is usual for employees to have at least two different emails. Each email address has the actual **email**, **type of email (work or personal).**
3. Employee phone number. One can have **at least one phone number**: personal phone, work phone number and a second, and third work phone number.

# Example

|  |  |  |
| --- | --- | --- |
| Employee | Email | Phone |
| John Avenger Ausini, Marketing, Junior Marketing Specialist, Pass.ID 643972016689 | Work: [ausini@marketing.bwct.com](mailto:ausini@marketing.bwct.com)  Personal: [john\_ausini@bestmail.top](mailto:john_ausini@bestmail.top) | Personal: +357123456789,  Work: +44357123456784,  Work: +45357123456783, |
| Pragubta Brahaman Kishivara, Programming, Senior Front-end developer, Pass. ID 43IN2432467545465 | Personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  Work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  Work: [summer\_vacations@programming.bwct.com](mailto:summer_vacations@programming.bwct.com) | Work: + (410)293-5762 |
| Hoordvig Van Derholdt,  Construction, Construction Worker,  Pass. ID H654364366546 | Work: [hoordvig@construction.bwct.com](mailto:hoordvig@construction.bwct.com)  Personal: [john\_ausini@bestmail.top](mailto:john_ausini@bestmail.top) | Work: + (31)293576245 |

# Todo

1.Create the structure of the **new tables** in **the geography database** because geographical data will be used further in the application.

2. Create three separate tables with proper names (like **employee, employee\_emails, employee\_phones**) with a primary key called **id** for each table.

3. Let the new tables have columns to relate them to one another. Table **employee\_email** needs an **employee\_id** foreign key. What about **employee\_phones**?

# Problem 2. HR Application Insert Employee

Janette needs first to insert the main employee data. Write the HR application as CLI with the following input and output:

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| John, Avenger, Ausini, Marketing, Junior Marketing Specialist, Pass 643972016689 | New employee John Avenger Ausini saved. |
| Pragubta, Brahaman, Kishivara, Programming, Senior Front-end developer, Pass 43IN2432467545465 | New employee Pragubt, Brahaman Kishivara saved. |
| Hoordvig, Van, Derholdt, Construction, Construction Worker, Pass H654364366546 | New employee Hoordvig Van Derholdt saved. |
| Hoordvig, Van, Derholdt, Construction Worker | Error: Please, check your input syntax. |

# Todo

1. Encapsulate the application in a class Employee.

|  |
| --- |
| File Employee.php:  <?php class Employee {  *//Todo* } |

2. Use a **db\_config.php** file which will hold your **database credentials** and initialize a connection using PDO kept in the **$db** variable. Set the error attribute to throw exceptions for MySQL errors. Do not catch exceptions!

|  |
| --- |
| File db\_config.php:  <?php $db\_host = "localhost"; $db\_name = "geography"; $db\_user = "root"; $db\_password = 'pass4yoursql'; $db = new PDO("mysql:dbname=$db\_name;host=$db\_host", $db\_user, $db\_password,  array(PDO::*ATTR\_ERRMODE* => PDO::*ERRMODE\_EXCEPTION*)); |

3.The entry point of your application should be **app.php**:

|  |
| --- |
| File app.php:  <?php include "db\_config.php"; include "Employee.php"; *// Todo* |

4. Pass **$db** wherever you need it **(like in the Employee class). Use Prepare / execute!**

5. Code the functionality to insert a new employee in the Employee class. Use an appropriate name of the method like insertEmployee().

7. **Insert the data** for the four employees in the **example** by **using your application** to test it.

# Problem 3. Insert Email

Now Janette needs to insert the emails of an employee. To do that she enters the three names of the employee. But there might be two employees with same names like Ivan Ivanov Ivanov:

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| Ivan, Ivanov, Ivanov, **emails**, **personal:** [ivanov@abv.bg](mailto:ivanov@abv.bg), **work:** [ivanov@bwct.com](mailto:ivanov@bwct.com), **work:** [summer\_deals@bwct.com](mailto:summer_deals@bwct.com) | Employees with this name: 123, 678 |
| **678**, Ivan, Ivanov, Ivanov, **emails**, **personal:** [ivanov@abv.bg](mailto:ivanov@abv.bg), **work:** [ivanov@bwct.com](mailto:ivanov@bwct.com), **work:** [summer\_deals@bwct.com](mailto:summer_deals@bwct.com) | Emails of Ivan Ivanov Ivanov saved. |
| Hoordvig, Van, Derholdt, **emails**, +7654656465655 | Error: Please, check your input email syntax. |

# Todo

1. Write a method to check if the person name is unique in the database. If the name is unique return an empty array else return their **IDs in an array.**

|  |
| --- |
| **// Unique: []**  **// Not unique: [123,456, 789,1234]** |

2.Use the method from point 1 and write a method to insert the emails**.**

If the employee is **not unique** print:

|  |
| --- |
| Employees with this name: 123,456,789,1234 |

Else print:

|  |
| --- |
| Emails of Ivan Ivanov Ivanov saved. |

3. Somewhere in the Employee class you should check the syntax. You may use a **main** or **validate** method to do that.

4.**Insert the emails** for the four employees from the example

# Problem 4. Insert Phones\*

To insert phone numbers we use pretty much the same like for the emails. The methodology for the uniqueness of the person should be used again. Try to implement the 3 steps from problem 3 by yourself to have the following input and output:

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| Ivan, Ivanov, Ivanov, **phones**, **personal:** +7654656465655, **work:** +7654656656565, **work:** +7654656656599 | Employees with this name: 123, 678 |
| Ivan, Ivanov, Ivanov, **phones**, **personal:** +7654656465655, **work:** +7654656656565, **work:** +7654656656599 | Phones of Ivan Ivanov Ivanov saved. |
| Hoordvig, Van, Derholdt, **phones**, work: derholdt@bwct.com | Error: Please, check your input phone number syntax. |

# Todo

Insert the phone numbers for the four employees from the example! Now in your database you should have complete data for four employees.

# Problem 5. Get Emails of Employee

When all employee data is entered Janette needs to extract the employees emails. She knows the employee **first name** and **last name.** Write **a method** like getInfo() that will show all employee data with their emails.

The Input and output should look like:

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| **Info**, Ivan, Ivanov | 123, Ivan, Ivanov, Ivanov, Marketing, Junior Marketer, **personal:** [ivanov@abv.bg](mailto:ivanov@abv.bg), **work:** [ivanov@bwct.com](mailto:ivanov@bwct.com), **work:** [summer\_deals@bwct.com](mailto:summer_deals@bwct.com)  - - - - - - - - - -  678, Ivan, Stoyanov, Ivanov, Marketing, Senior Marketer, **personal:** [ivanov\_is@abv.bg](mailto:ivanov_is@abv.bg), **work:** [is\_ivanov@bwct.com](mailto:is_ivanov@bwct.com) |
| **Info**, Pragubta, Kishivara | 3446, Pragubta Brahaman Kishivara, Programming, Senior Front-end developer, personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  work: summer\_vacations@programming.bwct.com |

Notice that if we have more than one person with the same first and last name we will **retrieve all employees** from the database **matching the names**. The **number (123, 678, 3446**) is the **id of the employee** which is important to Janette. The id of the employee is unique.

# Hint

Use a join of **employee** table with **employee\_emails** with **employee\_id** as a foreign key in **employee\_emails**.

# Problem 6. Get Emails and Phones\*

Add phone number functionality to the getInfo() method in the same manner like the output in problem 5.

# Part II. Altering Tables

# Problem 7. Country of Employees

The HRs needs to know the **native country** of each employee. So you need to add a **new column** **native\_country** to the employee table to hold the country code. But the native country might be **null** also and that is its default value. Do this by Heidi SQL. Observe the SQL query which is generated. Compare it with the alter SQL statements in the presentation.

# Todo

1.Write a new method getCountryCode() which returns from the database the two characters country code if such a country exists.

2.Change the method insertEmployee() to store the data of the **country code in employee** by using somewhere the new getCountryCode() method.

Then your input should look like this:

|  |
| --- |
| **Input** |
| John, Avenger, Ausini, Marketing, Junior Marketing Specialist, Pass 643972016689, **USA** |
| Pragubta, Brahaman, Kishivara, Programming, Senior Front-end developer, Pass 43IN2432467545465, **India** |
| Hoordvig, Van, Derholdt, Construction, Construction Worker, Pass H654364366546, **Netherlands** |

3.Change the method getInfo() to display the country code in the output:

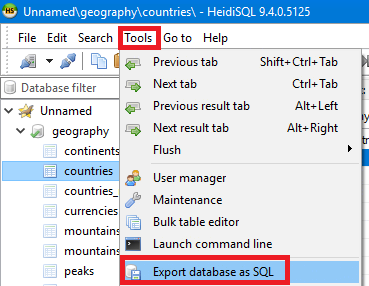
|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| **Info**, Ivan, Ivanov | 123, Ivan, Ivanov, Ivanov, Marketing, Junior Marketer, **from BG**, personal: [ivanov@abv.bg](mailto:ivanov@abv.bg), work: [ivanov@bwct.com](mailto:ivanov@bwct.com), work: [summer\_deals@bwct.com](mailto:summer_deals@bwct.com)  - - - - - - - - - -  678, Ivan, Stoyanov, Ivanov, Marketing, Senior Marketer, personal: [ivanov\_is@abv.bg](mailto:ivanov_is@abv.bg), work: [is\_ivanov@bwct.com](mailto:is_ivanov@bwct.com) |
| **Info**, Pragubta, Kishivara, | 3446, Pragubta Brahaman Kishivara, Programming, **from IN**, Senior Front-end developer, personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  work: summer\_vacations@programming.bwct.com |

# Problem 8. Show Native Country of Employees\*

Change **getInfo()** instead of country code to display the actual name of the country. Use getCountryCode().

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| **Info**, Ivan, Ivanov | 123, Ivan, Ivanov, Ivanov, Marketing, Junior Marketer, **from Bulgaria**, personal: [ivanov@abv.bg](mailto:ivanov@abv.bg), work: [ivanov@bwct.com](mailto:ivanov@bwct.com), work: [summer\_deals@bwct.com](mailto:summer_deals@bwct.com)  - - - - - - - - - -  678, Ivan, Stoyanov, Ivanov, Marketing, Senior Marketer, personal: [ivanov\_is@abv.bg](mailto:ivanov_is@abv.bg), work: [is\_ivanov@bwct.com](mailto:is_ivanov@bwct.com) |
| **Info**, Pragubta, Kishivara | 3446, Pragubta Brahaman Kishivara, Programming, **from India**, Senior Front-end developer, personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  work: summer\_vacations@programming.bwct.com |

# Problem 9. Alter Passport Id of Employees

Backup your whole database now (See the screenshot for HeidiSQL on the right).

Now alter the Passport Id column to hold only Varchar(3) type.

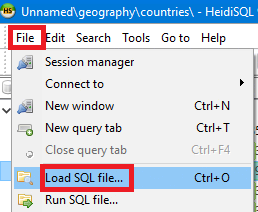
Observe now what happened with all entered data in the column!

|  |
| --- |
| **Important!** When you alter the tables in the database you can damage the data. This can happen when you change data types. Here all Passport Ids will be cut and so damaged. |

# 9.1 Show Passport Id

1. Change the method getInfo() to show Passport Ids. Restart your application to show them. How do they look like now?

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| **Info**, Pragubta, Kishivara | 3446, Pragubta Brahaman Kishivara, Programming, from IN, **Passport 43I** , Senior Front-end developer, personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  work: summer\_vacations@programming.bwct.com |



2.Drop the whole database

3.Restore the database from your backup file (see screenshot)

4.Restart the application. It should look like this:

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| **Info**, Pragubta, Kishivara | 3446, Pragubta Brahaman Kishivara, Programming, from IN, **Passport 43IN2432467545465** , Senior Front-end developer, personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  work: summer\_vacations@programming.bwct.com |

# Problem 10. Use Wildcards

Change the input interface to be able to use **wildcards** to show all employees starting with a specific combination of characters for their **first name** or **family name** like this:

|  |  |
| --- | --- |
| **Input** | **Ouptut** |
| **Info**, Prag\* | 3446, Pragubta Brahaman Kishivara, Programming, from IN, **Passport 43IN2432467545465** , Senior Front-end developer, personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  work: summer\_vacations@programming.bwct.com |
| **Info**, Kish\* | 3446, Pragubta Brahaman Kishivara, Programming, from IN, **Passport 43IN2432467545465** , Senior Front-end developer, personal: [keepcalmwhenucode@abv.bg](mailto:keepcalmwhenucode@abv.bg)  work: [pragubta@programming.bwct.com](mailto:pragubta@programming.bwct.com),  work: summer\_vacations@programming.bwct.com |
| **Info, Iv\*** | 123, Ivan, Ivanov, Ivanov, Marketing, Junior Marketer, **from Bulgaria**, personal: [ivanov@abv.bg](mailto:ivanov@abv.bg), work: [ivanov@bwct.com](mailto:ivanov@bwct.com), work: [summer\_deals@bwct.com](mailto:summer_deals@bwct.com)  - - - - - - - - - -  678, Ivan, Stoyanov, Ivanov, Marketing, Senior Marketer, personal: [ivanov\_is@abv.bg](mailto:ivanov_is@abv.bg), work: [is\_ivanov@bwct.com](mailto:is_ivanov@bwct.com) |

# Problem 11. Develop the Input Interface\*\*

Change the input interface to be able to insert **employee data, emails and phone numbers** altogether with one single command on the CLI. You can choose the format of input data yourself.

# 11.1 Add Update Methods

Add functionality to update employee data, email and phone numbers.

# 11.1 Add Delete Methods

Add functionality to delete employee with all their data. Add functionality do delete a single email or a single phone number for an employee.