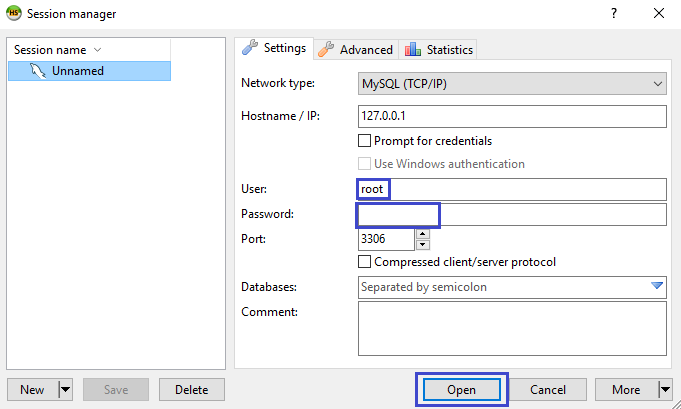
# Lab: Data Definition and Data Types

This document defines the **lab exercise assignments** for the **"Databases Basics - MySQL"** [course](https://softuni.bg/couses/database-basics-my-sql) **@ Software University**.

## Part 1. Simple Database Operations Using HeidiSQL

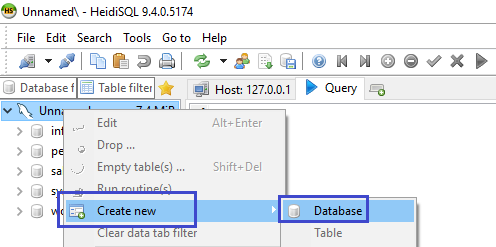
Before we start writing simple queries we are going to do some operations with the **GUI Client – HeidiSQL**. It offers us a very light and easy to use interface to create and alter database components.

Before we start we must connect to our instance with **root** and **root password**.

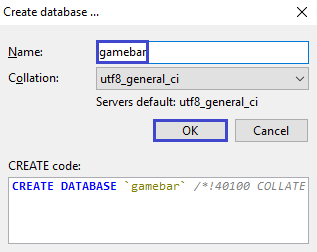


### 1. Create New Database

First, create an empty database gamebar. Right-click the instance you are working with and choose “**Create new**” -> “**Database**”.



New “**Create Database**” window will appear. In the “**Name**” field type the name of your new database – “**gamebar**”. The “Collation” menu will set the default collation for your database. **Don’t change anything there for now.**

****

In the “**CREATE code**” field you can see the query that is about to be executed.

### 2. Create New Table

Right click the “**gamebar**” database that you’ve created in the previous problem and select “**Create New**” -> “**Table**”.

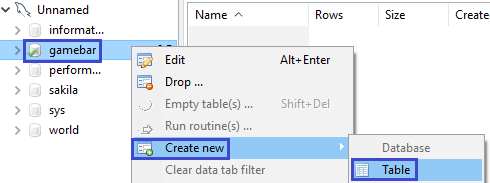
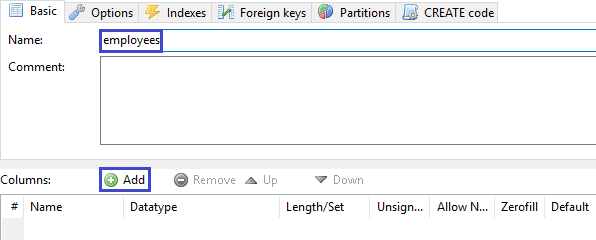
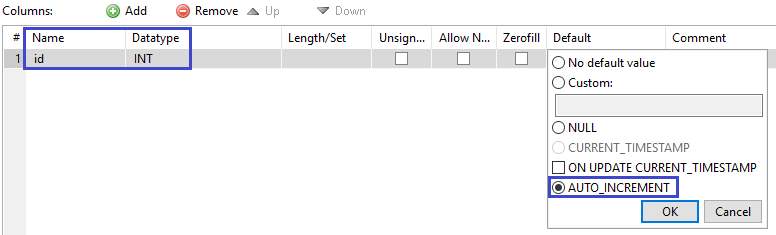


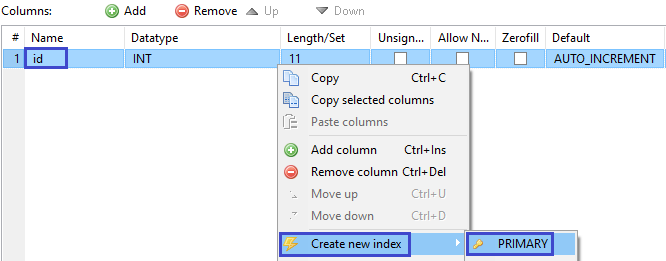
Table creation tab will appear. In the “**Name**” field type the name of your new table – “**employees**”. From the “**Add**” button you can start creating your table fields.



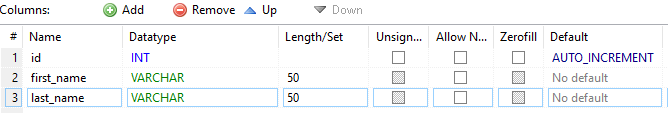
First create an “**id**” field. It will be set to INT and AUTO\_INCREMENT. Select AUTO\_INCREMENT from the “**Default**” field.



Make the “**id**” field to be primary key.



Create 2 more fields – “**first\_name**” and “**last\_name**”.



Similar to “**employees**” create 2 more tables.

Table “**categories**”:

• id – INT, primary key, AUTO\_INCREMENT;

• name – VARCHAR, NOT NULL;

Table “**products**”:

• id – INT, primary key, AUTO\_INCREMENT;

• name – VARCHAR, NOT NULL;

• category\_id – INT, foreign key referenced to the “**categories**” table (id)

**Foreign keys** are created in the “**Foreign keys**” tab. Click the “**Add**” button and specify:

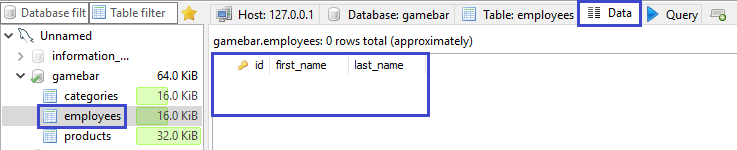
• **Columns** – select the column you want to be set as foreign key – “category\_id”;

• **Reference** **table** – select the table from which you will choose a column to link your foreign key – “categories”;

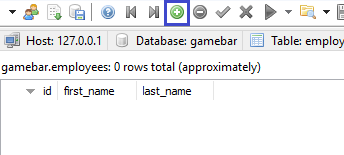
• **Foreign** **columns** – select the column set to primary to link the foreign key – “id”;

### 3. Insert Data in Tables

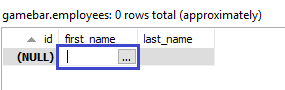
Now we can start adding some records to our newly created tables. First select the “**employees**” table. From the “**Data**” tab you can see all the records that are already inserted. Initially the table is empty.



Select the **green** **plus** **button** to add new record.

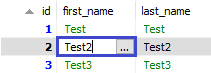


Fill in the fields with values. Create 3 records in each table.



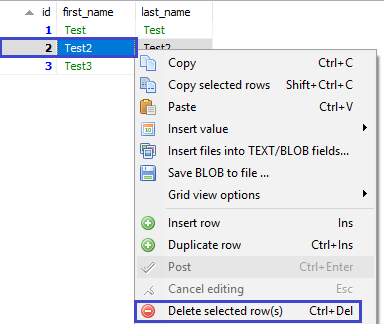
### 4. Editing Data

Data in tables can easily be edited with the GUI. Now that we’ve populated our tables with test records we can edit them by **clicking on the value field**.



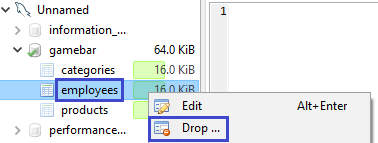
### 5. Deleting Data

Data deletion is easy too. We just right click the row we want to delete and select “**Delete selected row(s)**”.



### 6. Dropping Tables

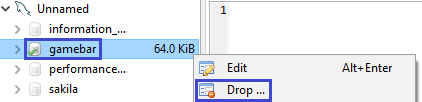
We can delete the whole table, by selecting the one we want to delete, right click and choose “**Drop…**”. **You cannot undo this action.**



### 7. Dropping the Database

As table dropping, we can drop the database too. **This action cannot be undone too.**

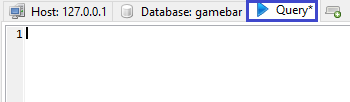
Right click the database you want to drop and select “**Drop…**”.



## Part 2. Simple Database Operations Using Queries

Now we are going to do the same steps from Part 1 using simple MySQL queries.

Queries are written in the “Query” tab.



### 1. Create New Database

Write a query that will create the “**gamebar**” database.

### 2. Create Tables

When we create tables, we specify the database we want to add them to. This is done by using the “USE” clause.

**Submit your solutions without the** “USE {database name}” **row.**

Create the “**categories**” and “**products**” tables analogically.

Table “**categories**”:

• id – INT, primary key, AUTO\_INCREMENT;

• name – VARCHAR, NOT NULL;

Table “**products**”:

• id – INT, primary key, AUTO\_INCREMENT;

• name – VARCHAR, NOT NULL;

• category\_id – INT, NOT NULL;

### 3. Insert Data in Tables

Inserting data can be done with a query too. To do that we use the “INSERT” clause. Populate the “**employees**” table with 3 test values.

### 4. Altering Tables

Altering the tables is done via the “ALTER TABLE” clause. Add a new column – “**middle\_name**” to the “**employees**” table.

### 5. Adding Constraints

Create the connection via foreign key between the “**products**” and “**categories**” tables that you’ve created earlier. Make “**category\_id**” **foreign key linked to “id” in the “categories” table**.

### 6. Modifying Columns

Change the property “VARCHAR(50)” to “VARCHAR(100)” to the “**middle\_name**” column in “**employees**” table.

### 7. Drop Database

Drop the “**gamebar**” database.