# Lab: Data Definition and Data Types

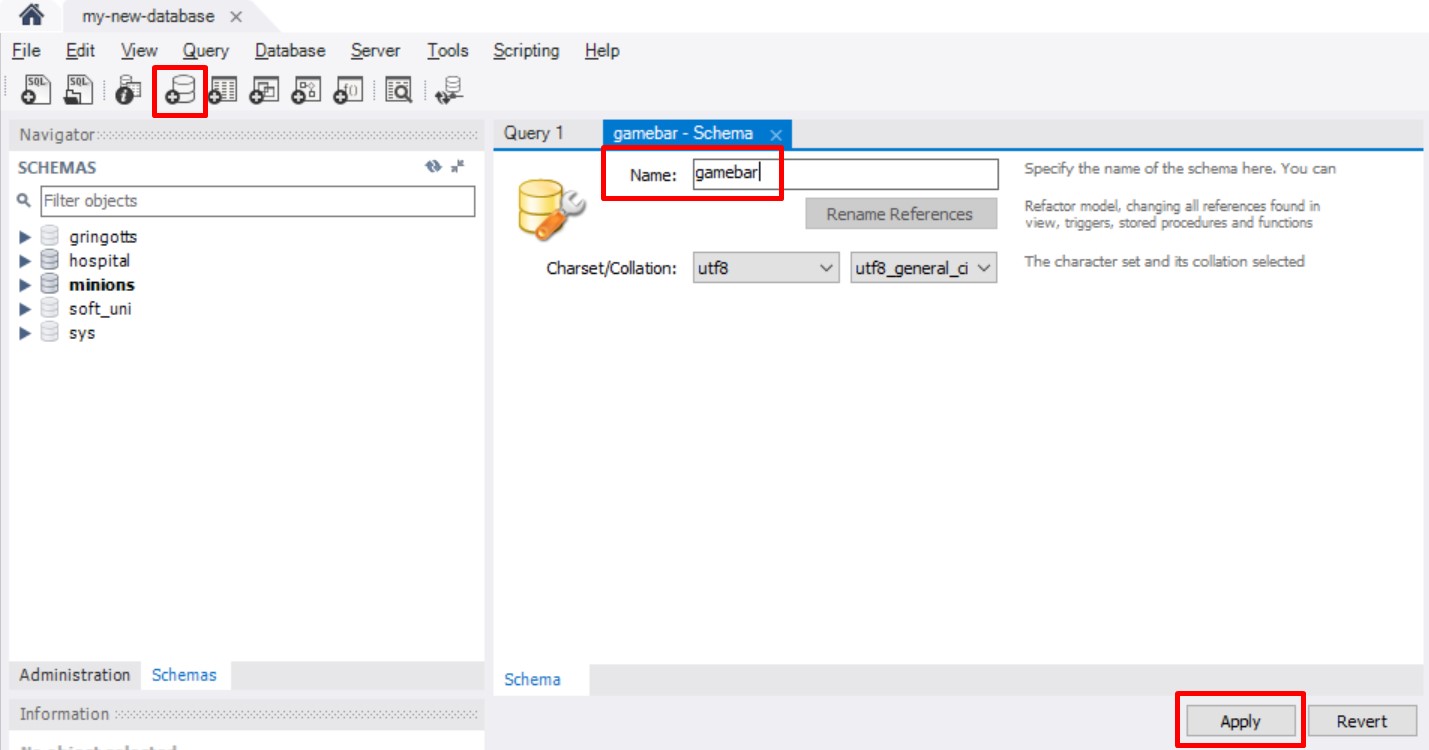
This document defines the **lab exercise assignments** for the [**MySQL course @ Software University**](https://softuni.bg/opencourses/databases-basics-mysql).

## Simple Database Operations Using MySQL Workbench

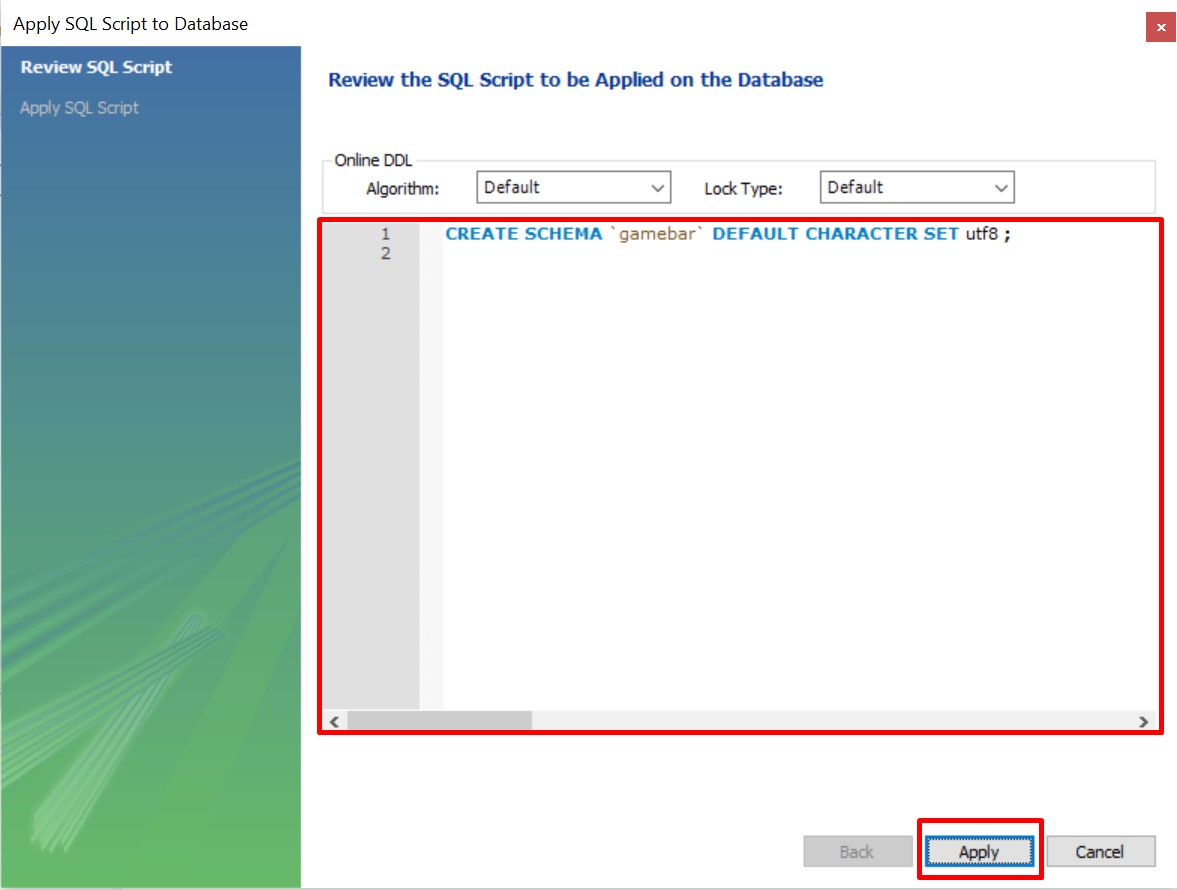
### Create New Database

First, create an empty database gamebar.

New Create **Database** window will appear. In the "**Name**" field type the name of your new   
database - "**gamebar**".



After clicking "Apply" you can see new window with the query that is about to be executed.



### Create New Table

Right click the "**Tables**" and select "**Create Table**".

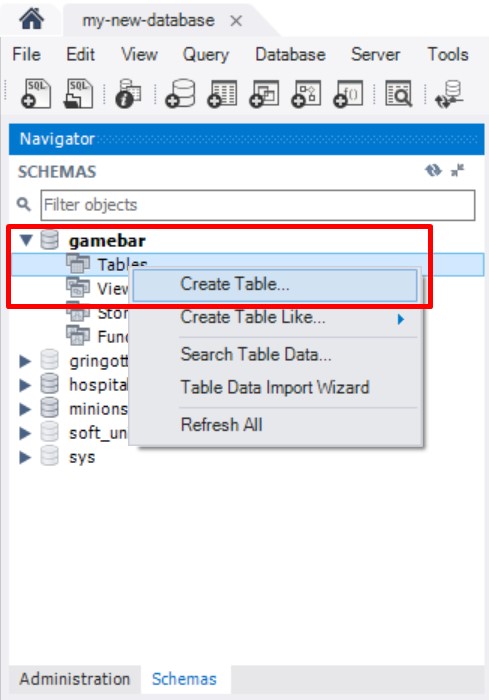
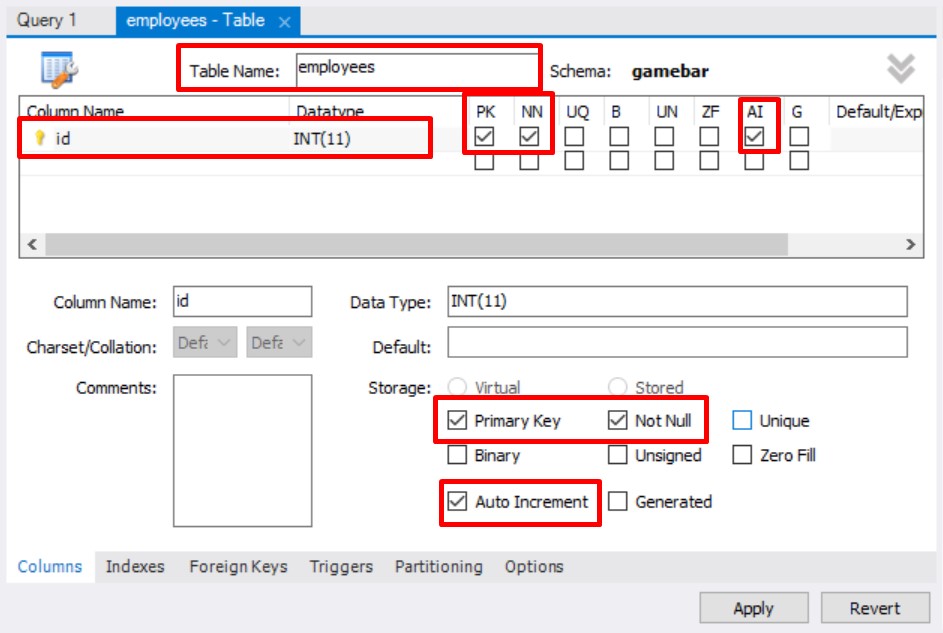


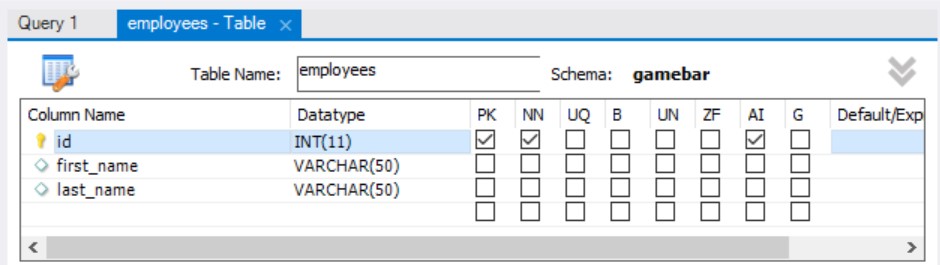
Table creation tab will appear. In the "**Name**" field type the name of your new table – "**employees**". From the "**Columns**" tab you can start creating your table fields.

First create an "**id**" field. It will be set to INT and AUTO\_INCREMENT. Select Auto Increment (AI).

Make the "**id**" field to be Primary Key (PK).



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;

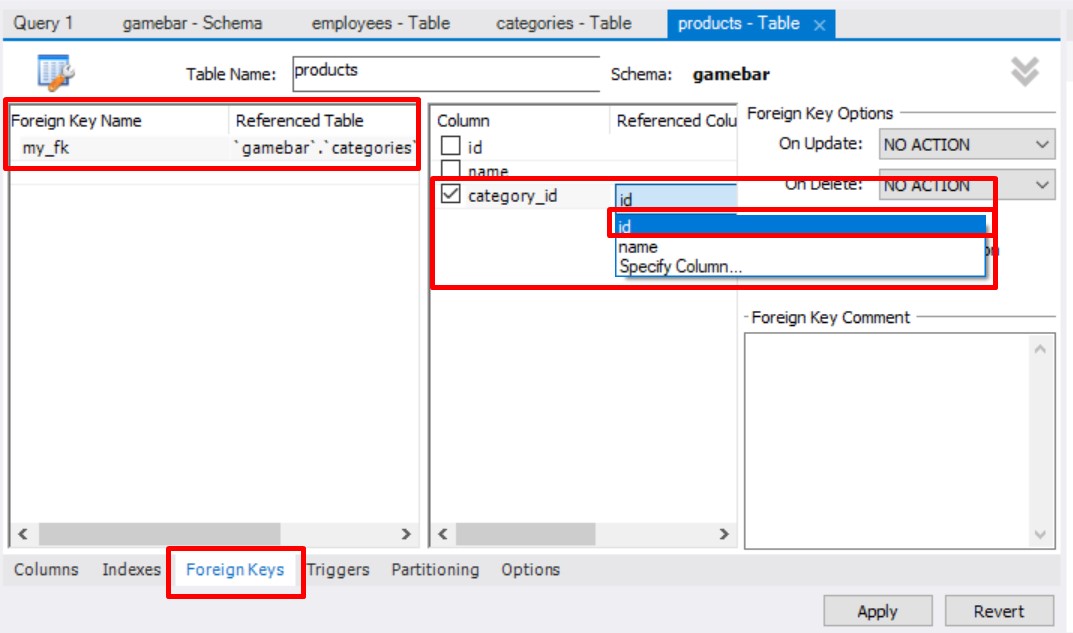
• category\_id – INT, foreign key referenced to the "**categories**" table (id)

**Foreign keys** are created in the "**Foreign keys**" tab:

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

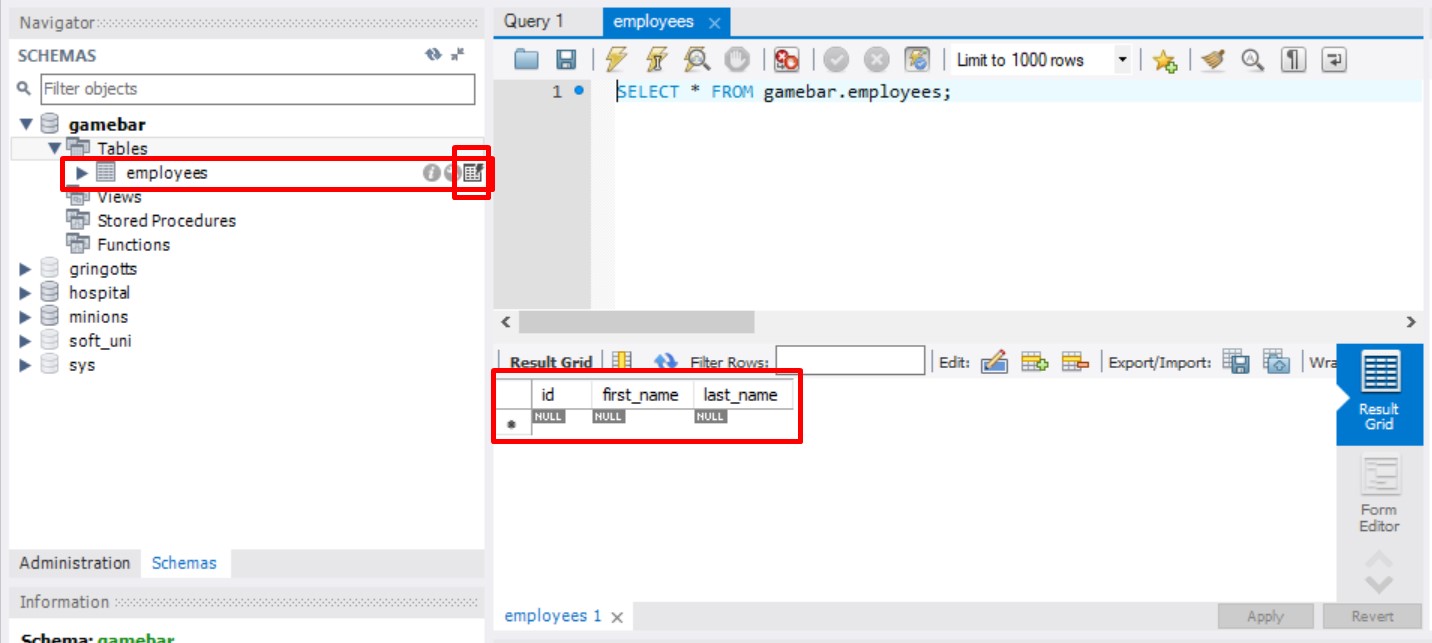
* **Columns** – select the column you want to be set as foreign key – "category\_id";

• **Referenced** **columns** – select the column set to primary to link the foreign key – "id";



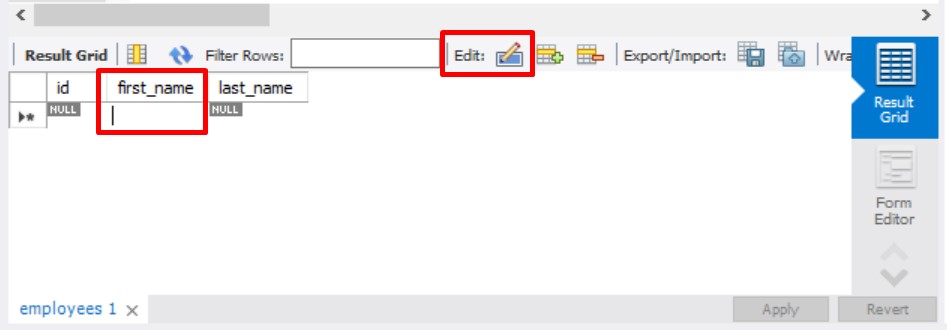
### Insert Data in Tables

Now we can start adding some records to our newly created tables. First select the "**employees**" table:



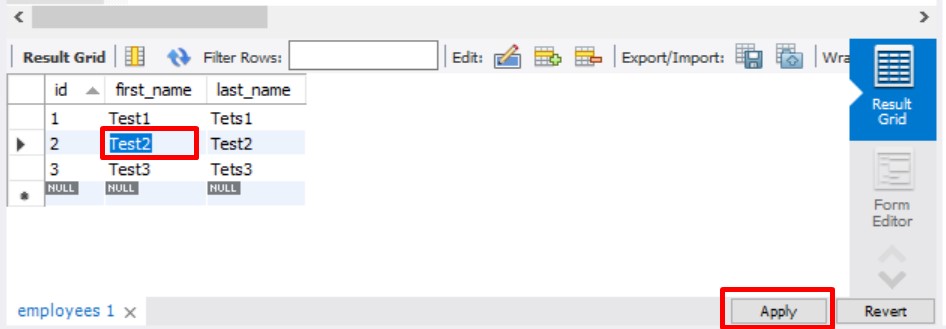
Select the **Edit** **button** to add new record.

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



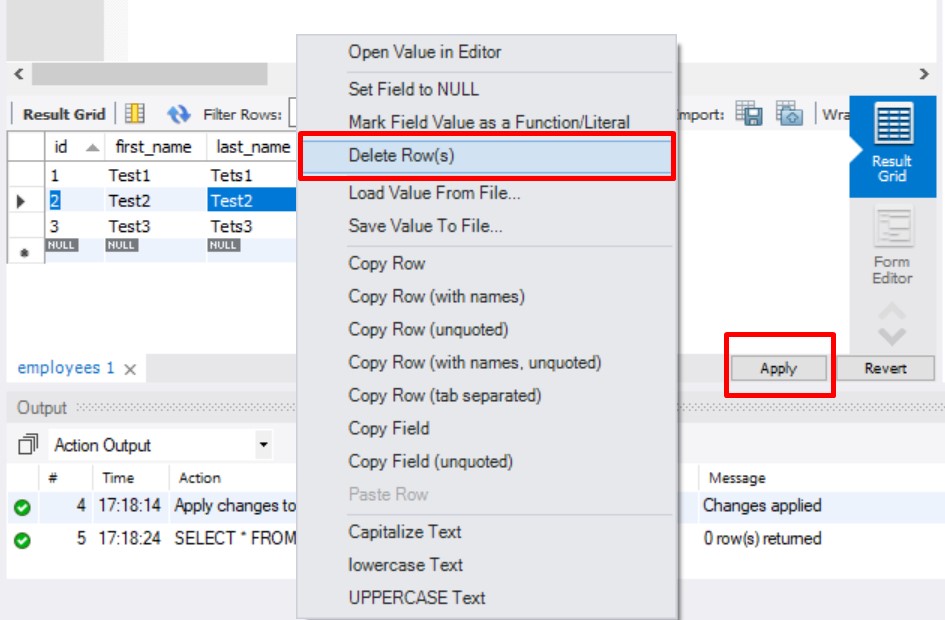
### Editing Data

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**.



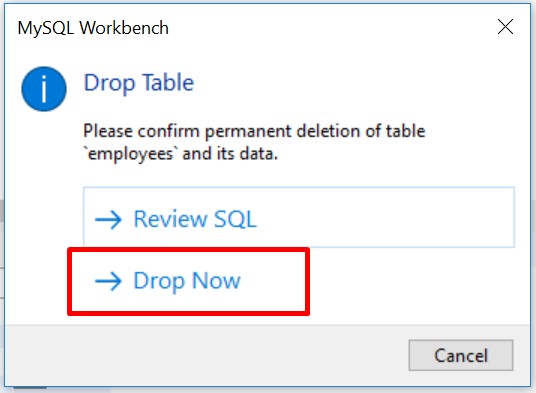
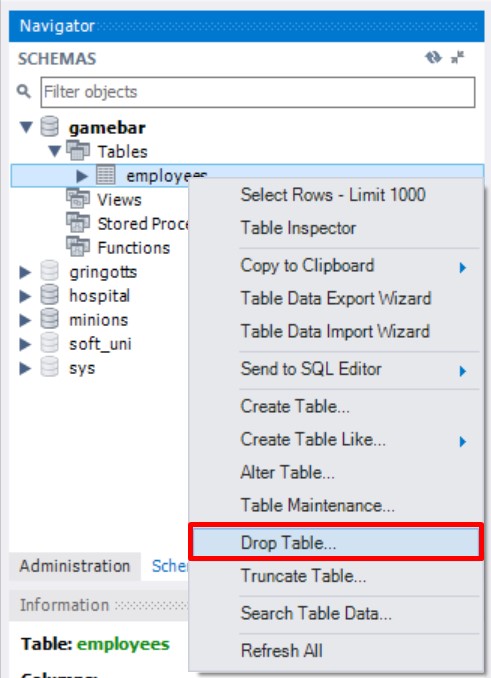
### Deleting Data

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



### Dropping Tables

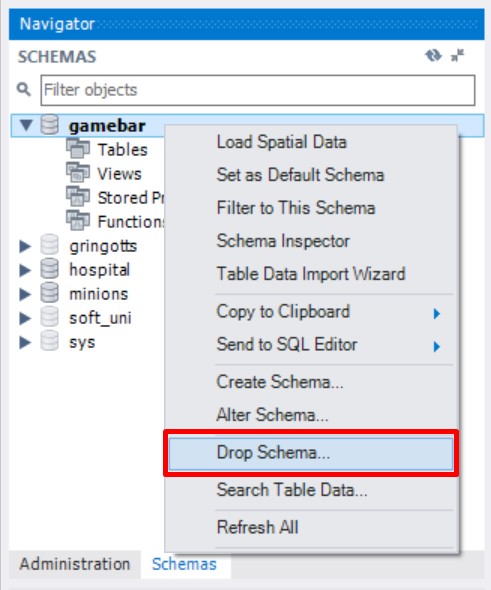
We can delete the whole table, by selecting the one we want to delete, right click and choose "**Drop Table…**". **You cannot undo this action.**



### 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 Schema…**".

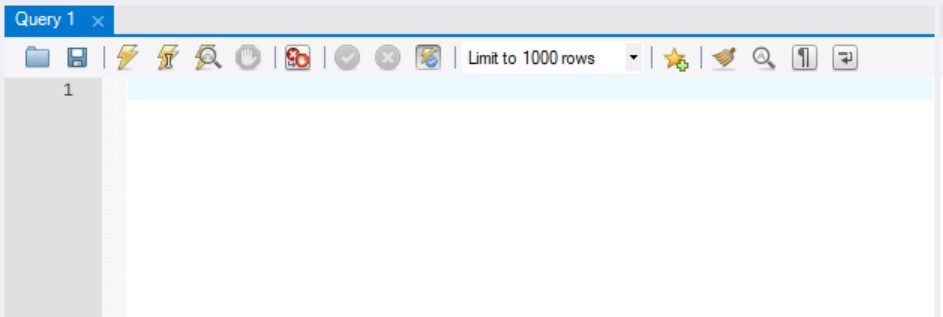


## Simple Database Operations Using Queries

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

Exercises from this section should be submit in JUDGE – From 1 to 5.

Queries are written in the "Query" tab.



### 0. Create New Database

Write a query that will create the "**gamebar**" database.

### 1. 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 in JUDGE without the** "USE {database name}" **row.**

Table "**employees**":

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

• first\_name – VARCHAR, NOT NULL;

• last\_name – VARCHAR, NOT NULL;

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 – it is not a foreign key for now.

### 2. 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.

### 3. Alter Tables

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

### 4. 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**.

### 5. Modifying Columns

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