# Exercises: Table Relations

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

## One-To-One Relationship

Create two tables as follows. Use appropriate data types.

|  |  |  |  |
| --- | --- | --- | --- |
| **persons** | | | |
| **person\_id** | **first\_name** | **salary** | **passport\_id** |
| 1 | Roberto | 43300.00 | 102 |
| 2 | Tom | 56100.00 | 103 |
| 3 | Yana | 60200.00 | 101 |

|  |  |
| --- | --- |
| **passports** | |
| **passport\_id** | **passport\_number** |
| 101 | N34FG21B |
| 102 | K65LO4R7 |
| 103 | ZE657QP2 |

**Insert the data from the example above.**

* Alter table **persons** and make person\_ida primary key.
* Create a foreign key between **persons** and **passports** by using the passport\_id column.
* Think about which passport field should be UNIQUE.

Submit your queries by using “**MySQL run queries & check DB**” strategy**.**

## One-To-Many Relationship

Create two tables as follows. Use appropriate data types.

|  |  |  |
| --- | --- | --- |
| **manufacturers** | | |
| **manufacturer\_id** | **name** | **established\_on** |
| 1 | BMW | 01/03/1916 |
| 2 | Tesla | 01/01/2003 |
| 3 | Lada | 01/05/1966 |

|  |  |  |
| --- | --- | --- |
| **models** | | |
| **model\_id** | **name** | **manufacturer\_id** |
| 101 | X1 | 1 |
| 102 | i6 | 1 |
| 103 | Model S | 2 |
| 104 | Model X | 2 |
| 105 | Model 3 | 2 |
| 106 | Nova | 3 |

**Insert the data from the example above.**

* Add primary and foreign keys.

Submit your queries by using “**MySQL run queries & check DB”** strategy.

## Many-To-Many Relationship

Create three tables as follows. Use appropriate data types.

|  |  |
| --- | --- |
| **students** | |
| **student\_id** | **name** |
| 1 | Mila |
| 2 | Toni |
| 3 | Ron |

|  |  |
| --- | --- |
| **exams** | |
| **exam\_id** | **name** |
| 101 | Spring MVC |
| 102 | Neo4j |
| 103 | Oracle 11g |

|  |  |
| --- | --- |
| **students\_exams** | |
| **student\_id** | **exam\_id** |
| 1 | 101 |
| 1 | 102 |
| 2 | 101 |
| 3 | 103 |
| 2 | 102 |
| 2 | 103 |

**Insert the data from the example above.**

* Add primary and foreign keys.
* Have in mind that the table student\_exams should have a **composite** primary key.

Submit your queries by using “**MySQL run queries & check DB**” strategy**.**

## Self-Referencing

Create a single table as follows. Use appropriate data types.

|  |  |  |
| --- | --- | --- |
| **teachers** | | |
| **teacher\_id** | **name** | **manager\_id** |
| 101 | John |  |
| 102 | Maya | 106 |
| 103 | Silvia | 106 |
| 104 | Ted | 105 |
| 105 | Mark | 101 |
| 106 | Greta | 101 |

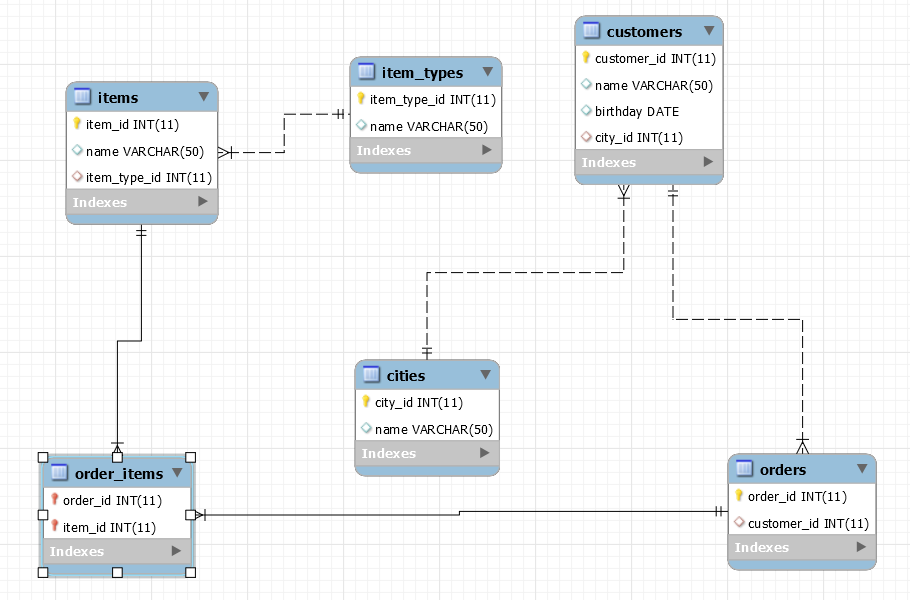
**Insert the data from the example above.**

* Add primary and foreign keys.
* The foreign key should be between manager\_id and teacher\_id.

Submit your queries by using “ **MySQL run queries & check DB**” strategy**.**

## Online Store Database

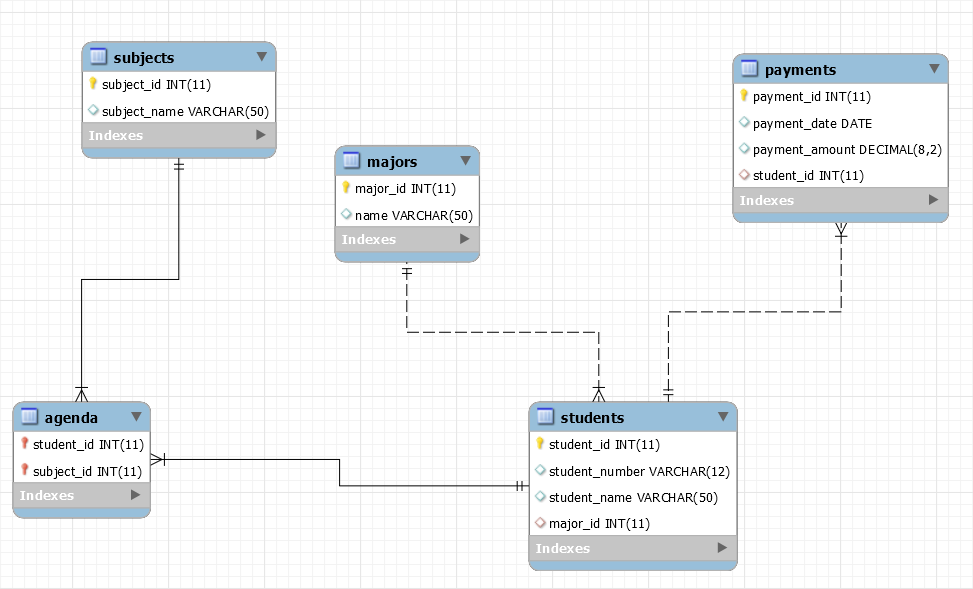
Create a new database and design the following structure:



Submit your queries by using “**MySQL run queries & check DB”** strategy**.**

## University Database

Create a new database and design the following structure:



Submit your queries by using “**MySQL run queries & check DB**” strategy.

## SoftUni Design

Create an E/R Diagram of the SoftUni Database. There are some special relations you should check out: **employees** are **self-referenced** **(manager\_id)** and **departments** have **One-to-One** with the **employees** **(manager\_id)** while the **employees** have **One-to-Many** (**department\_id**). You might find it interesting how it looks on а diagram. ☺

## Geography Design

Create an E/R Diagram of the Geography Database.

## Peaks in Rila

Display all peaks for "**Rila**" mountain\_range. Include:

* mountain\_range
* peak\_name
* peak\_elevation

Peaks should be sorted by peak\_elevation descending.

### Example

|  |  |  |
| --- | --- | --- |
| **mountain\_range** | **peak\_name** | **peak\_elevation** |
| Rila | Musala | 2925 |
| … | … | … |

Submit your queries by using “**MySQL prepare DB & run queries**” strategy.