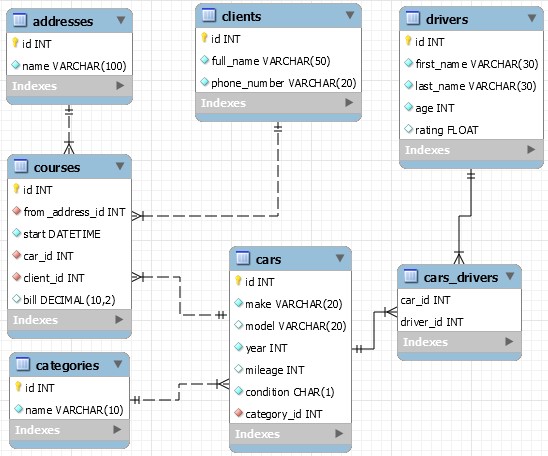
# MySQL Exam SoftUni Taxi Company

As part of the best students in SoftUni, you are charged with the difficult task of developing an example database for another new direction of SoftUni - a taxi company. Since you are not so familiar with this industry you will be given a document with detailed explanations of what this database should be. You will also receive testing data, with which you will be able to do many tests to prove that you have managed to fully cope with your task.

## Section 0: Database Overview

You have been given an Entity / Relationship Diagram of the SoftUni Taxi Company:



The **SoftUni Taxi Company (stc)** needs to hold information about **cars**, **courses**, **drivers**, **clients, addresses and categories**.

Your task is to create a database called stc (**SoftUni Taxi Company**). Then you will have to create several **tables**:

* cars – contains information about the **cars**
  + Each car has a make column, a model column, a year column, a mileage column, a condition column and a category column
* courses – contains information about the **courses**
  + Each course has a from\_address column, a start column, a car column, a client column and a bill column
* drivers – contains information about the **drivers**
  + Each driver has a first and last name columns, an age column and a rating column
* clients – contains information about the **clients**
  + Each client has a full name column and a phone numbercolumn
* addresses – contains information about the **addresses**
* categories – contains information about the **categories**
  + Contains the name of the category
* cars\_drivers – a **many** to **many** **mapping** tables between the **cars** and the **drivers**
  + Have composite primary key from the **car\_id** column and the **driver\_id** column

## Section 1: Data Definition Language (DDL) – 40 pts

Make sure you implement the whole database correctly on your local machine so that you can work with it.

The instructions you will be given will be the minimal required for you to implement the database.

### Table Design

You have been tasked to create the tables in the database by the following models:

#### addresses

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **100 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |

#### categories

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**, from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **10 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |

#### clients

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**, from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| full\_name | A **string** containing a maximum of **50 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| phone\_number | A **string** containing a maximum of **20 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |

#### drivers

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| first\_name | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| last\_name | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| age | **Integer,** from **1** to **2,147,483,647.** | **NULL** is **NOT** permitted**.** |
| rating | **Floating point** number | **DEFAULT value is 5.5**  **NULL** is permitted. |

#### cars

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| make | A **string** containing a maximum of **20 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| model | A **string** containing a maximum of **20 characters**. Unicode is **NOT** needed. | **NULL is permitted.** |
| year | **Integer**, from **1** to **2,147,483,647.** | **DEFAULT value is 0**  **NULL** is **NOT** permitted**.** |
| mileage | **Integer**, from **1** to **2,147,483,647.** | **DEFAULT value is 0**  **NULL** is permitted. |
| condition | **character** that shows the **condition** of the car. One character. | **NULL** is **NOT** permitted**.** |
| category\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table categories.  **NULL** is **NOT** permitted**.** |

#### courses

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer**, from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| from\_address\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table addresses.  **NULL** is **NOT** permitted**.** |
| start | The **date** and **time** when the course starts | **NULL** is **NOT** permitted. |
| bill | **DECIMAL**, up to **10 digits**, **2** of which after the **decimal point**. | **DEFAULT value** is 10 |
| car\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table cars.  **NULL** is **NOT** permitted**.** |
| client\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table clients.  **NULL** is **NOT** permitted**.** |

#### cars\_drivers

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| car\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table cars.  **NULL** is **NOT** permitted**.** |
| driver\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table drivers.  **NULL** is **NOT** permitted**.** |

* + Have composite primary key from the **car\_id** column and the **driver\_id** column

Submit your solutions in Judge on the first task. Submit **all** SQL table creation statements.

You will also be given a data.sql file. It will contain a **dataset** with data that you will need to **store** in your **local database**. This data will be given to you, so you do not have to imagine it and lose precious time in the process.  
The data is in the form of **INSERT** statement queries.

## Section 2: Data Manipulation Language (DML) – 30 pts

Here we need to do several manipulations in the database, like changing data, adding data etc.

### Insert

When drivers are not working and need a taxi to transport them, they will also be registered   
at the database as customers.

You will have to **insert** records of data into the **clients** table, based on the **drivers** table.

For all drivers with an id between 10 and 20(**both** **inclusive**), **insert data** in the clients table with the **following values**:

* full\_name –get the first and last name of the driver separated by a single space
* phone\_number – set it to start with **(088) 9999 and** the driver\_id multiplied by 2
  + Example – the phone\_number of the driver with id = 10 is (088) 9999**20**

### Update

After many kilometers and over the years, the condition of cars is expected to deteriorate.

Update all **cars** and set the **condition** to **'C'**. The cars must have a **mileage** greater than **800000 (inclusive)** or **NULL** and must be older than **2010**(**inclusive**).

**Skip** the cars that contain a **make** value of **Mercedes-Benz**. They can work for many more years.

### Delete

Some of the clients have not used the services of our company recently, so we need to remove them   
from the database.

**Delete** all clients from clientstable**,** that do not have any courses and the **count** of the **characters** in the **full\_name** is more than **3** characters**.**

## Section 3: Querying – 50 pts

And now we need to do some data extraction. **Note** that the **example results** from **this section** use a **fresh database**. It is **highly recommended** that you **clear** the **database** that has been **manipulated** by the **previous problems** from the **DML** **section** and **insert again** the **dataset** you have been given, to ensure **maximum consistency** with the **examples** given in this section.

### Cars

Extract the info about all the **cars**.

**Order** the results by **car’s id.**

#### Required Columns

* make
* model
* condition

#### Example

|  |  |  |
| --- | --- | --- |
| **make** | **model** | **condition** |
| Land Rover | Range Rover | A |
| GMC | Sierra 3500 | A |
| Infiniti | IPL G | B |
| … | … | … |
| Ford | E250 | C |
| Infiniti | EX | C |

### Drivers and Cars

Now, we need more detailed information about drivers and their cars.

Select all drivers and cars that they drive. Extract the driver’s **first** and **last** name from the driver's table and the **make**, **model** and **mileage** from the cars table. **Order** the result by the **mileage** in **descending** order, then by the **first** name alphabetically.

**Skip** all cars that have **NULL** as a value for the **mileage**.

#### Required Columns

* first\_name
* last\_name
* make
* model
* mileage

#### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **first\_name** | **last\_name** | **make** | **model** | **mileage** |
| Edna | Heatley | Pontiac | Trans Sport | 967608 |
| Delaney | Stove | Pontiac | Fiero | 959778 |
| Cristi | Ravenshear | Maserati | GranTurismo | 954606 |
| … | … | … | … | … |
| Ailina | Sebyer | Mercedes-Benz | G-Class | 6482 |
| Gerhard | Alderson | Mercedes-Benz | G-Class | 6482 |

### Number of courses for each car

Extract from the database all the **cars** and the **count** of their **courses**. Also display the **average** **bill** of each course by the **car**, rounded to the second digit.

**Order** the results **descending** bythe **count** of **courses,** then bythe **car’s id**.

**Skip** the cars with **exactly** **2** courses.

#### Required Columns

* car\_id
* make
* mileage
* count\_of\_courses
* avg\_bill

#### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **car\_id** | **make** | mileage | count\_of\_courses | avg\_bill |
| 13 | Mercedes-Benz | 846549 | 4 | 26.95 |
| 80 | Lincoln | 711184 | 4 | 22.39 |
| 1 | Land Rover | 550259 | 3 | 14.71 |
|  | … | … | … | … |
| 89 | Pontiac | 890015 | 0 | NULL |
| 90 | Mercedes-Benz | 6482 | 0 | NULL |
| 95 | Land Rover | 176967 | 0 | NULL |

### Regular clients

Extract the **regular clients, who have ridden in more than one car**. The **second** **letter** of the customer's **full name** must be **'a'**. Select the **full name**, the **count** of **cars** that he has ridden and the total **sum** of all courses.

**Order** clients by their **full\_name**.

#### Required Columns

* full\_name
* count\_of\_cars
* total\_sum

#### Example

|  |  |  |
| --- | --- | --- |
| **full\_name** | **count\_of\_cars** | **total\_sum** |
| Haven Seaton | 4 | 129.65 |
| Jacquelynn Plackstone | 2 | 64.36 |
| Kaylee Coushe | 4 | 135.71 |
| Lanita Crockatt | 2 | 84.12 |
| Parker McGeorge | 2 | 62.58 |
| Randie Cridge | 5 | 154.65 |
| Raynor Dobbison | 2 | 66.77 |

### Full information on courses

The headquarters want us to make a query that shows the complete information about all courses in the database. The information that they need is the **address**, if the course is made in the **Day** (between **6** and **20**(inclusive both)) or in the **Night** (between **21** and **5**(inclusive both)), the **bill** of the course, the **full name** of the **client**, the car **maker**, the **model** and the **name** of the **category**.

Order the results by **course id.**

#### Required Columns

* name (address)
* day\_time
* bill
* full\_name (client)
* make
* model
* category\_name (category)

#### Example

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **name** | day\_time | bill | full\_name | make | model | category\_name |
| 2 Del Mar Park | Night | 46.92 | Randie Cridge | Mitsubishi | Galant | Hatchback |
| 24402 Warner Place | Day | 14.47 | Henrik Sivyer | Lincoln | MKT | Hatchback |
| 80678 Green Ridge Court | Day | 44.01 | Randie Cridge | Isuzu | Rodeo Sport | Cabrio |
| … | … | … | … | … | … | … |
| 34 Briar Crest Crossing | Night | 14.86 | Georges Lanston | Isuzu | Rodeo Sport | Cabrio |
| 86462 Buena Vista Point | Day | 16.06 | Jefferson Montacute | Buick | Hearse | Coupe |

## Section 4: Programmability – 30 pts

The time has come for you to prove that you can be a little more dynamic on the database. So, you will have to write several procedures.

### Find all courses by client's phone number

Create a **user defined function** with the name **udf\_courses\_by\_client (phone\_num VARCHAR (20))** that receives a **client’s phone number** and returns the number of courses that clients have in the database.

#### Example

|  |
| --- |
| **Query** |
| SELECT udf\_courses\_by\_client ('(803) 6386812') as `count`; |
| count |
| **5** |

|  |
| --- |
| **Query** |
| SELECT udf\_courses\_by\_client ('(831) 1391236') as `count`; |
| count |
| **3** |

|  |
| --- |
| **Query** |
| SELECT udf\_courses\_by\_client ('(704) 2502909') as `count`; |
| count |
| **0** |

### Full info for address

Create a stored procedure udp\_courses\_by\_address which accepts the following parameters:

* address\_name (with max length 100)

Extract data about the **addresses** with the given **address\_name.** The needed data is the **name** of the **address**, the **full name** of the **client**, the **level of a bill** (depends on the course bill – **Low** – lower than **20**(inclusive), **Medium** – lower than **30**(inclusive), and **High**), **make** and **condition** of the car and the **name** of the **category**.

**Order** addresses by **make,** then by client’s **full** **name**.

#### Required Columns

* name (address)
* full\_name
* level\_of\_bill
* full\_name (client)
* make
* condition
* cat\_name (category)

|  |
| --- |
| **Query** |
| CALL udp\_courses\_by\_address('700 Monterey Avenue'); |

#### Result

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **name** | **full\_names** | **level\_of\_bill** | **make** | **condition** | **cat\_name** |
| 700 Monterey Avenue | Kelcy Cody | Medium | Acura | B | Hatchback |
| 700 Monterey Avenue | Zeke Rowston | Medium | GMC | A | Coupe |
| 700 Monterey Avenue | Joyann Garrettson | High | Lamborghini | A | SUV |
| 700 Monterey Avenue | Courtney Gawkes | Low | Mercedes-Benz | B | Cabrio |
| 700 Monterey Avenue | Jeralee Tue | Low | Mercedes-Benz | B | Cabrio |
| 700 Monterey Avenue | Haven Seaton | High | Mitsubishi | B | Hatchback |

|  |
| --- |
| **Query** |
| CALL udp\_courses\_by\_address('66 Thompson Drive'); |

#### Result

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **name** | **full\_names** | **level\_of\_bill** | **make** | **condition** | **cat\_name** |
| 66 Thompson Drive | Kimball Deem | High | Pontiac | C | Hatchback |
| 66 Thompson Drive | Kaylee Coushe | High | Porsche | B | Coupe |
| 66 Thompson Drive | Gibbie Liggens | High | Volkswagen | A | Coupe |