# PostgreSQL Exam Preparation II

Exam problems for the [PostgreSQL course @ Software University](https://softuni.bg/trainings/4244/postgresql-september-2023).

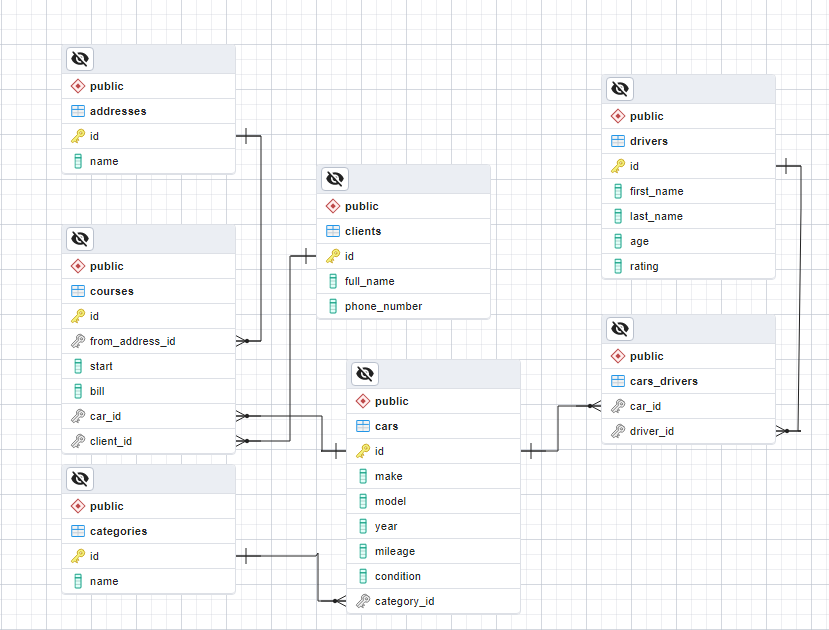
**Submit your solutions** in the SoftUni [Judge Contest](https://judge.softuni.org/Contests/4297/Exam-Prep-II).

# Taxi Company

*As one of the top-performing students at SoftUni, you have been assigned the challenging responsibility of designing a sample database for a new venture - a taxi company. Despite being less familiar with this industry, you will be provided with a comprehensive document outlining the requirements for this database. Additionally, you will receive test data that will enable you to conduct thorough testing and demonstrate your successful completion of the task.*

# Section 1. Data Definition Language (DDL) - (30 pts)

You have been provided with an E/R Diagram depicting the structure of the **Taxi Company**.



Create a PostgreSQL database named **"taxi\_db"** and design the following tables:

* **"addresses"** - contains information about the addresses;
* **"categories"** - stores information about the categories;
* **"clients"** - stores information about the clients;
* **"drivers"** - holds information about the drivers;
* **"cars"** - contains information about the cars.
* **"courses"** - stores information about the courses.
* **"cars\_drivers"** - serves as a many-to-many mapping table between cars and drivers.

**NOTE: Please ensure that you use the exact data types specified in the model tables when working with dates. For instance, if a column is of type** **"DATE," make sure to use the "DATE" type. Similarly, if a column is of type "TIMESTAMP" use the "TIMESTAMP" data type. Failure to use the correct data types may result in the Judge system rejecting your submission as incorrect.**

**Furthermore, it's important to keep in mind that foreign keys should adhere to the following naming convention:**

**fk\_<referencing\_table>\_<referenced\_table>**

##### Your first assignment is to create the database tables based on the provided models. Follow the given specifications to create the tables**.** Ensure that the constraints match the order of columns.

##### **addresses**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **name** | **String** up to **100** symbols | **NULL** is **not** allowed |

##### **categories**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **name** | **String** up to **10** symbols | **NULL** is **not** allowed |

##### **clients**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **full\_name** | **String** up to **50** symbols | **NULL** is **not** allowed |
| **phone\_number** | **String** up to **20** symbols | **NULL** is **not** allowed |

##### **drivers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **first\_name** | **String** up to **30** symbols | **NULL** is **not** allowed |
| **last\_name** | **String** up to **30** symbols | **NULL** is **not** allowed |
| **age** | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Must be a positive number |
| **rating** | **Numeric** number with two-digit precision | The **DEFAULT** value is **5.5**  **NULL** is permitted |

##### **cars**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **make** | **String** up to **20** symbols | **NULL** is **not** allowed |
| **model** | **String** up to **20** symbols | **NULL** is permitted |
| **year** | **Integer** from **0** to **2,147,483,647** | The **DEFAULT** value is **0**, the column must always have a value **greater than**  **NULL** is **not** allowed, |
| **mileage** | **Integer** from **0** to **2,147,483,647** | The **DEFAULT** value is **0**, the column must always have a value **greater than**  **NULL** is permitted |
| **condition** | **String** limited to **1** character | **NULL** is **not** allowed |
| **category\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **categories**, Cascade Operations, **NULL** is **not** allowed |

##### **courses**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| from\_address\_id | **Integer** from **0** to **2,147,483,647** | Relationship with table **addresses**, Cascade Operations, **NULL** is **not** allowed |
| start | **TIMESTAMP** when the course starts | **NULL** is **not** allowed |
| bill | **Numeric** number with ten-digit precision and two-digit scale | The **DEFAULT** value is **10**, the column must always have a value **greater than**  **NULL** is permitted |
| car\_id | **Integer** from **0** to **2,147,483,647** | Relationship with table **cars**, Cascade Operations,  **NULL** is **not** allowed |
| client\_id | **Integer** from **0** to **2,147,483,647** | Relationship with table **clients**, Cascade Operations, **NULL** is **not** allowed |

##### **cars\_drivers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| car\_id | **Integer** from **0** to **2,147,483,647** | Relationship with table **cars**, Cascade Operations,  **NULL** is **not** allowed |
| driver\_id | **Integer** from **0** to **2,147,483,647** | Relationship with table **drivers**, Cascade Operations,  **NULL** is **not** allowed |

## Database Design

Submit only your **CREATE** statements for all tables to the Judge.

# Section 2. Data Manipulation Language (DML) - (10 pts)

**Prior to beginning, it is necessary to import "dataset.sql". If the structure has been properly created, the data should be inserted successfully.**

In this scenario, you have multiple tasks that involve manipulating the database, such as modifying data and adding new entries.

## Insert

When drivers require transportation and are not working, they should be registered as customers in the database. Your task is to insert records into the **"clients"** table based on the **"drivers"** table. Specifically, for drivers with an **"id"** **between** **10** and **20** (**inclusive**), you need to insert data into the **"clients"** table with the following values:

* **"full\_name"** - combine the **"first\_name"** and **"last\_name"** of the driver, separated by a **single space**;
* **"phone\_number"** -set it to begin with **"(088) 9999"** followed by the driver's **"id"** **multiplied** by **2**. For instance, if the driver's **"id"** is **10**, the phone number should be **"(088) 999920"**.

### Example

|  |  |  |
| --- | --- | --- |
| **id** | **full\_name** | **phone\_number** |
| 1 | Kalindi Keningham | (704) 2502909 |
| 2 | Lois Leidl | (933) 4279635 |
| 3 | Casi Farens | (933) 4275652 |
| … | … | … |
| 99 | Marsha Cridge | (230) 6839868 |
| 100 | Elysha Maydwell | (842) 4057110 |
| 101 | Delaney Stove | (088) 999920 |
| 102 | Ilaire Tomaszewicz | (088) 999922 |
| 103 | Genna Jaquet | (088) 999924 |
| 104 | Carlotta Dykas | (088) 999926 |
| … | … | … |
| 110 | Roddie Gribben | (088) 999938 |
| 111 | Boyce Briddock | (088) 999940 |

## Update

To account for the wear and tear that occurs over time and long distances, it is necessary to update the condition of **"cars"** in the database. All cars meeting the following criteria should be updated to have a **"condition"** of **'C'**:

* the cars must have a **"mileage"** **greater than** **800,000** (**inclusive**) or have a **NULL** value;
* the cars must be **older than or equal** to **2010**;
* cars with a **"make"** value of **'Mercedes-Benz'** should be **skipped**, as they are expected to continue functioning well.

### Example

Before update

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **id** | **make** | **model** | **year** | **mileage** | **condition** | **category\_id** |
| 1 | Land Rover | Range Rover | 2021 | 550259 | A | 3 |
| 2 | GMC | Sierra 3500 | 2012 | 275538 | A | 4 |
| 3 | Infiniti | IPL G | 2004 | [null] | B | 1 |
| … | … | … | … | … | … | … |
| 18 | Infiniti | G35 | 2004 | [null] | A | 2 |
| … | … | … | … | … | … | … |
| 26 | Volkswagen | Jetta | 2002 | 345452 | C | 3 |
| 27 | Aston Martin | Vantage | 2010 | 890984 | C | 4 |
| … | … | … | … | … | … | … |
| 36 | Mitsubishi | Pajero | 2005 | 921496 | B | 4 |
| … | … | … | … | … | … | … |

After update

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **id** | **make** | **model** | **year** | **mileage** | **condition** | **category\_id** |
| 1 | Land Rover | Range Rover | 2021 | 550259 | A | 3 |
| 2 | GMC | Sierra 3500 | 2012 | 275538 | A | 4 |
| 3 | Infiniti | IPL G | 2004 | [null] | C | 1 |
| … | … | … | … | … | … | … |
| 18 | Infiniti | G35 | 2004 | [null] | C | 2 |
| … | … | … | … | … | … | … |
| 26 | Volkswagen | Jetta | 2002 | 345452 | C | 3 |
| 27 | Aston Martin | Vantage | 2010 | 890984 | C | 4 |
| … | … | … | … | … | … | … |
| 36 | Mitsubishi | Pajero | 2005 | 921496 | C | 4 |
| … | … | … | … | … | … | … |

## Delete

As part of the assignment, you are tasked with removing **"clients"** from the database who have not recently used the services of the company. The criteria for removal are as follows: delete all clients from the clients table who do **not have any associated courses** and have a **"full\_name"** **length greater than 3** characters.

### Example

Before delete

|  |  |  |
| --- | --- | --- |
| **id** | **full\_name** | **phone\_number** |
| 1 | Kalindi Keningham | (704) 2502909 |
| 2 | Lois Leidl | (933) 4279635 |
| 3 | Casi Farens | (933) 4275652 |
| 4 | Janna Kellert | (353) 9465732 |
| … | … | … |
| 10 | Meris Shale | (842) 4350411 |
| 11 | Colline Dann | (698) 8299305 |
| 12 | Joyann Garrettson | (858) 8642667 |
| 13 | Shane Arr | (261) 2901780 |
| 14 | Owen Strivens | (807) 1053029 |
| … | … | … |

After delete

|  |  |  |
| --- | --- | --- |
| **id** | **full\_name** | **phone\_number** |
| 2 | Lois Leidl | (933) 4279635 |
| 4 | Janna Kellert | (353) 9465732 |
| … | … | … |
| 12 | Joyann Garrettson | (858) 8642667 |
| 14 | Owen Strivens | (807) 1053029 |
| … | … | … |

# Section 3. Querying - (40 pts)

**Now we will perform some data extraction tasks. Please note that the example results provided in this section are based on a fresh database. It is highly recommended to clear the database that was manipulated in the previous problems from the DML section and insert the given dataset again to ensure consistency with the examples in this section.**

## Cars

Extract information about all the **"cars"**, including their **"make"**, **"model"**, and **"condition"**. Sort the results in ascending order based on the car's **"id"**.

**Example**

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

## Drivers and Cars

#### Retrieve comprehensive information about drivers and their cars. Extract the **"first\_name"** and **"last\_name"** of each driver, along with the **"make"**, **"model"**, and **"mileage"** of their cars. Arrange the results in **descending order** of **"mileage"**. If there are any drivers with the same mileage, sort them **alphabetically** by **"first\_name"**. **Exclude** any cars that have a **NULL** value for **"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

Your task is to write an SQL query that retrieves information from the database regarding all the cars and the count of their courses. Additionally, you need to display the **"average\_bill"** of each course for each car, **rounded to the second decimal digit**. The results should be ordered in **descending order** based on the **"count\_of\_courses"**. In case multiple cars have the same number of courses, the results should be further ordered by the car's **"id"**. You should **exclude** cars that have **exactly 2 courses** from the results.

**Example**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **car\_id** | **make** | mileage | count\_of\_courses | average\_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 details of regular clients who have traveled in **multiple cars**. Filter the clients based on the condition that the **second letter** of their **full name** is **'a'**. Retrieve the **"full\_name"**, and **"count\_of\_cars"** they have traveled in and the **"total\_sum"** of their course **bills**. Sort the result by the **"full\_names"** of the clients.

**Example**

|  |  |  |
| --- | --- | --- |
| **full\_name** | **count\_of\_cars** | **total\_sum** |
| Haven Seaton | 4 | 129.65 |
| Jacquelynn Plackstone | 2 | 64.36 |
| Kaylee Coushe | 4 | 135.71 |
| … | … | … |
| Raynor Dobbison | 2 | 66.77 |

## Full Information of Courses

The headquarters has requested a query to retrieve comprehensive information about all courses in the database. The required information includes the **"address"**, whether the course was conducted during the **'Day'** (**between 6 and 20, inclusive**) or the **'Night'**(**between 21 and 5, inclusive**), the course **"bill"**, the **"full\_name"** of the client, the car **"make"**, **"model"**, and the **"category\_name"**. The results should be ordered by the course **"id"**.

**Example**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **address** | 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 - (20 pt)

## Find all Courses by Client’s Phone Number

As an assignment, your task is to create a user-defined function named **fn\_courses\_by\_client()** that accepts a client's phone number as a parameter (**"phone\_num"** of type **VARCHAR(20)**) and returns the **number of courses** that the client has in the database.

### For this task, please only submit your user-defined function in the Judge system.

**Example**

|  |  |
| --- | --- |
| **Query** | **Output** |
| **SELECT fn\_courses\_by\_client('(803) 6386812')** | **5** |
| **SELECT fn\_courses\_by\_client('(831) 1391236')** | **3** |
| **SELECT fn\_courses\_by\_client('(704) 2502909')** | **0** |

## Full Info for Address

You are required to create a stored procedure named **sp\_courses\_by\_address()** that accepts a parameter **"address\_name"** with a **maximum length of 100**. The procedure should retrieve information about addresses that match the given **"address\_name"**. The extracted data should include the **"address\_name"**, client's **"full\_name"**, the **"level\_of\_bill"** (categorized as **'Low'** if the bill is less than or equal to **20**, **'Medium'** if it is less than or equal to **30**, and **'High'** otherwise), the car's **"make"** and **"condition"**, and the **"category\_name"**. The results should be ordered by the car's **"make"**. In case there are multiple courses made by the same car, the results should be further sorted by the client's **"full\_name"**.

**\*\*\*** Please be aware that to view the procedure's results in a tabular format and conduct efficient testing within the Judge System, it's crucial to establish a table named **"search\_results"**. This table will serve as a container for the data generated by your stored procedure. Prior to creating the procedure, itself, execute the subsequent SQL query to create the **"search\_results"** table:

**CREATE TABLE search\_results (**

**id SERIAL PRIMARY KEY,**

**address\_name VARCHAR(50),**

**full\_name VARCHAR(100),**

**level\_of\_bill VARCHAR(20),**

**make VARCHAR(30),**

**condition CHAR(1),**

**category\_name VARCHAR(50)**

**);**

### Before beginning the insertion process into the table, ensure that you TRUNCATE any existing results from the table at the start of the procedure.

### In this task, please ensure that you only submit your stored procedure and the SQL query to create the table in the Judge system.

### Example

|  |
| --- |
| **Query** |
| **CALL usp\_courses\_by\_address('700 Monterey Avenue')**  **SELECT \* FROM search\_results;** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Output | | | | | |
| **address\_name** | full\_name | level\_of\_bill | make | condition | category\_name |
| 700 Monterey Avenue | Kelcy Cody | Medium | Acura | B | Hatchback |
| … | … | … | … | … | … |
| 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 usp\_courses\_by\_address('66 Thompson Drive')**  **SELECT \* FROM search\_results;** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Output | | | | | |
| **address\_name** | full\_name | level\_of\_bill | make | condition | category\_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 |