**Course title:** Data base 420-921-VA

**Project title:** Car2Go

**Instructor name:** Jaina Sheth

**Student name:** Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan

**Date:** 23-06-2022

**Table of contents:**

[**Introduction:** 3](#_Toc106981054)

[**Team Work Distribution** 4](#_Toc106981055)

[**Scenario description** 5](#_Toc106981056)

[**Conceptual Design of the Database** 6](#_Toc106981057)

[**Logical Database Schema** 7](#_Toc106981058)

[**Functional Dependencies and Database Normalization** 9](#_Toc106981059)

[**Database Tables** 10](#_Toc106981060)

[**Challenges and Suggestions for improvements** 15](#_Toc106981061)

[**Conclusions and Future Work** 16](#_Toc106981062)

[**References** 17](#_Toc106981063)

# **Introduction:**

This report is about the design of the management system database for a car rental company called Car2Go. The different possibilities of the presented scenario were examined to access the database presented in this report.

The first step in designing the Car2Go database was the conceptual design of the database which includes the complete Entity-Relationship (ER) model of our database, including tables listing entities involved in relationships along with relationship labels and connectivity.

The second step was to prepare the database schema, which represents the logical configuration of all parts of the ER diagram that we achieved from the conceptual design of the database. Then we listed the relationship schemas of the database along with a table listing foreign keys, referenced relationships, and referencing relationships.

The third step was to list the functional dependencies identified for each relationship that we got from the logical schema of the database. In the normalization process, all tables were in normal forms 1NF, 2NF and 3NF.

The fourth step was to create a database with tables according to the relational schema using DDL statements that support all constraints. All tables have been successfully implemented.

The last step was to use the DML statements, populate the tables with a reasonable amount of data, and print the contents of each table after populating the tables No error message received, and implementation was successful.

Since this was our first database design experience, we encountered some challenges and doubts during the implementation of this project. These challenges required a lot of teamwork consultations, which was a valuable experience for all team members. Overall, the design of the database was the most challenging step for all of us.

# **Team Work Distribution**

To practice and being experienced in database designing, our team members made the decision that each team member would do each step of the project individually and then combine all the tasks together. It was a good decision we made, because all the team members get experience in all steps of Car2Go project as well as discussing and benefiting of all members way of thinking. This project was a mindset sharing for us and a pleasant experience of a teamwork project.

|  |  |  |
| --- | --- | --- |
|  | Tasks | Done by |
| 1 | Description | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 2 | Business rules or Assumptions | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 3 | Conceptual Design - ERD | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 4 | Logical Design - Relational Schema | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 5 | Normalization | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 6 | DDL | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 7 | DML | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 8 | Queries\_script.sql | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |
| 9 | Project Report | Alexander Gutkovsky, Zahra Mirzaei, Enes Karaaslan |

# **Scenario description**

A company named Car2Go runs a rental business in several locations with different addresses. Car2Go provides subcompacts, compacts, sedans, or luxury car options with different properties. The company rents the car with full tank and records the volume of gas in the tank when the car is returned as well as the odometer reading before renting and after returning the car. The rented car in a particular location may be returned to a different location without extra charges. A customer can rent only one car at any given time and if the company does not have the customer requested car in stock, may provide a higher car class with the price of the customer’s requested car class. For certain weeks in the year, the company has promotional rentals for only a car class. Cra2Go keeps the record of customers. Calculation of prices are based on car classes, duration of renting days as well as promotion percentage if applicable.

1. The location table stores information about branches.

* Location\_id is unique (Primary Key) in location table to uniquely identify each row of the table

1. The location rent keeps record of the location in which car rented and returned

* Rent\_id as well as location\_start are unique in location\_rent relation

1. The car stores information about particular properties of the cars

* plate is unique in car relation

1. The car options are classified and car class keeps record of car options

* class\_name is unique in car\_class relation

1. The customer table keeps information about the customers

* driving\_license\_no is unique in customer table

1. The customer phone keeps record of the customers’ phone number

* customer\_ph is unique in customer\_ph table

1. The customer email keeps record of the customers’ email

* customer\_email is unique in customer\_email relation

1. The rent keeps records of rented car by the customer

* rent\_id is unique in rent relation

1. The promotion keeps record of the given yearly promotions

* promo\_id is unique in promotion table

1. The reservation keeps record of total rented days as well as price

* res\_id is unique in reservation relation

# **Conceptual Design of the Database**

Diagram

Description automatically generated

# **Logical Database Schema**

* customer (driving\_licence\_no, f\_name, l\_name, street, street\_no, apt, city, province, postal\_code, country)
* customer\_ph(driving\_licence\_no, country\_code,  area\_code, local\_number)
* customer\_email(driving\_licence\_no, email)
* location(location\_id, street\_name, street\_no, city, province\_branch, postal\_code\_branch)
* car (plate, vin, year, make, model, color, class\_name, location\_id)
* rent(rent\_id, driving\_licence\_no, st\_date, end\_date, gas\_tank, int\_odo, final\_odo, requested\_car\_class, rented\_car\_class, plate )
* location\_rent(rent\_id, location\_start, location\_final)
* car\_class( class\_name, cost\_per\_day)
* promotion(promo\_id, discount, class\_name, start\_date, last\_date)
* reservation(res\_id, total\_days(), price(),rent\_id, promo\_id)

Yellow represents foreign key.

Diagram

Description automatically generated with medium confidence

# **Functional Dependencies and Database Normalization**

Customer relation:

**driving\_licence\_no->** f\_name, l\_name, street, street\_no, apt, city, province, postal\_code, country

Location relation:

**location\_id-->** street\_name, street\_no, city, province\_branch, postal\_code\_branch

Car relation:

**Plate-->**vin, year, make, model, color, class\_name, location\_id

Rent relation:

**rent\_id-->** driving\_licence\_no, st\_date, end\_date, gas\_tank, int\_odo, final\_odo, requested\_car\_class, rented\_car\_class, plate

Location\_rent relation:

**rent\_id, location\_start-->** location\_final

Car\_class relation:

**class\_name-->**cost\_per\_day

Promotion relation:

**promo\_id-->**discount, class\_name, start\_date, last\_date

Reservation relation:

**res\_id-->**total\_days(), price (),rent\_id, promo\_id

*(--> represents dependency, example: A-->B means, B depends on A)*

**NORMALIZATION:**

1. All the tables are in 1NF.   
2. All the tables are in 2NF.   
3. All the tables are in 3NF.

# **Database Tables**

Table: Location

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| LOCATION\_ID | VARCHAR(5) |
| STREET\_NAME | VARCHAR(30) |
| STEET\_NO | INT |
| CITY | VARCHAR(30) |
| PROVINCE\_BRANCH | VARCHAR(30) |
| POSTAL\_CODE\_BRANCH | VARCHAR(20) |
| **Constraint** | CONSTRAINT PK\_LOCATION\_ID primary key(LOCATION\_ID) |

Graphical user interface, text, application

Description automatically generated

Table: Car

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| PLATE | CHAR(9) |
| VIN | CHAR(17) |
| YEAR | INT |
| MAKE | VARCHAR(20) |
| MODEL | VARCHAR(20) |
| COLOR | VARCHAR(15) |
| CLASS\_NAME | VARCHAR(30) |
| LOCATION\_ID | VARCHAR(5) |
| **Constraint** | CONSTRAINT FK\_CAR01 FOREIGN KEY(CLASS\_NAME) REFERENCES CAR\_CLASS (CLASS\_NAME) |
| CONSTRAINT FK\_CAR02 FOREIGN KEY(LOCATION\_ID) REFERENCES LOCATION (LOCATION\_ID) |
| CONSTRAINT PK\_CAR PRIMARY KEY(PLATE) |

Graphical user interface, application, Word

Description automatically generated

Table: Car\_Class

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| CLASS\_NAME | VARCHAR(30) |
| COST\_PER\_DAY | MONEY |
| **Constraint** | CONSTRAINT PK\_CAR\_CLASS PRIMARY KEY(CLASS\_NAME) |

Graphical user interface, application

Description automatically generated

Table: Promotion

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| promotion\_id | VARCHAR(20) |
| discount | DECIMAL(3,2) |
| class\_name | VARCHAR(30) |
| st\_date | date |
| lt\_date | date |
| **Constraint** | constraint pk\_promtion primary key(promotion\_id) |
| Constraint fk\_promotion01 foreign key (class\_name) references car\_class(class\_name) |

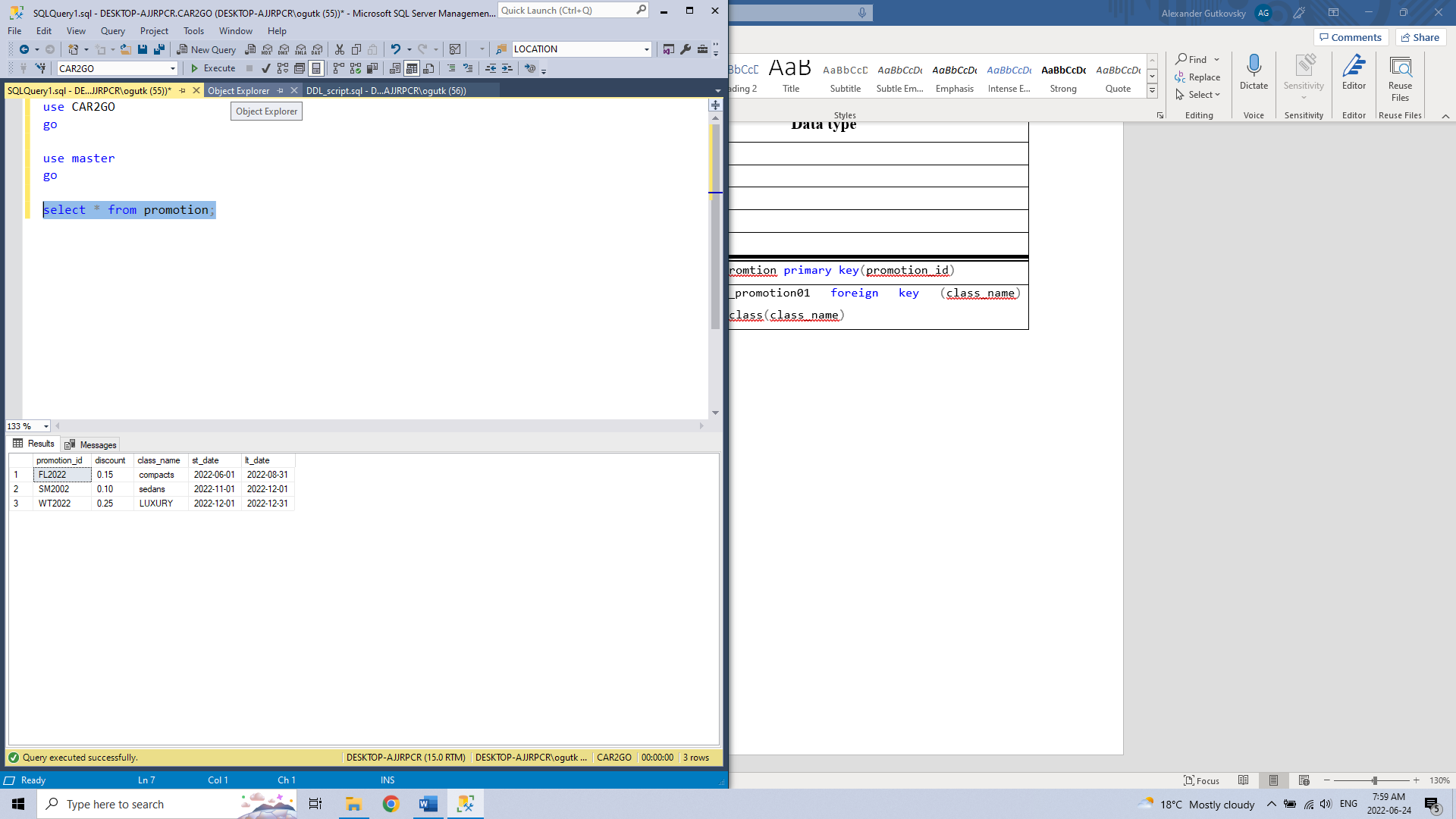


Table: Customer

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| driving\_license\_no | VARCHAR(20) |
| f\_name | VARCHAR(30) |
| l\_name | VARCHAR(30) |
| street\_name | VARCHAR(30) |
| street\_no | int |
| apt | int |
| city | VARCHAR(30) |
| province | VARCHAR(30) |
| postal\_code | VARCHAR(30) |
| country | VARCHAR (30) |
| **Constraint** | constraint  pk\_driving\_license\_no primary key( driving\_license\_no) |

Graphical user interface, application

Description automatically generated

Table: Customer\_ph

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| country\_code | int |
| area\_code | int |
| local\_code | int |
| driving\_license\_no | VARCHAR (20) |
| **Constraint** | constraint  pk\_customer\_ph primary key(driving\_license\_no, country\_code, area\_code, local\_code) |
| constraint fk\_customer\_ph foreign key(driving\_license\_no) references customer(driving\_license\_no) |

Graphical user interface, application, Word

Description automatically generated

Table: Custome\_email

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| email | VARCHAR(30) |
| driving\_licence\_no | VARCHAR(20) |
| **Constraint** | Constraint pk\_customer\_email primary key(email,driving\_licence\_no) |
| constraint fk\_customer\_email foreign key(driving\_licence\_no) references customer(driving\_license\_no) |

Graphical user interface, application, Word

Description automatically generated

Table: Rent

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| rent\_id | VARCHAR(20) |
| st\_date | DATE |
| end\_date | DATE |
| gas\_tank | VARCHAR(15) |
| int\_odo | int |
| final\_odo | int |
| requested\_car\_class | VARCHAR(30) |
| rented\_car\_class | VARCHAR(30) |
| plate | CHAR(9) |
| driving\_license\_no | VARCHAR (20) |
| **Constraint** | constraint fk\_rent02 foreign key(driving\_license\_no) REFERENCES customer (driving\_license\_no) |
| constraint fk\_rent01 foreign key( plate) REFERENCES car (plate) |
| constraint fk\_rent03 foreign key(requested\_car\_class) references car\_class(class\_name) |
| constraint fk\_rent04 foreign key(rented\_car\_class) references car\_class(class\_name) |

Graphical user interface, application, Word

Description automatically generated

Table: Location\_rent

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| location\_start | VARCHAR(5) |
| location\_final | VARCHAR(5) |
| rent\_id | VARCHAR (20) |
| **Constraint** | constraint pk\_location\_rent primary key(rent\_id,location\_start) |
| constraint fk\_location\_rent foreign key(rent\_id) references rent(rent\_id) |

Graphical user interface, application, Word

Description automatically generated

Table: Reservation

|  |  |
| --- | --- |
| **Attribute** | **Data type** |
| res\_id | VARCHAR(20) |
| total\_days | int |
| price | MONEY |
| rent\_id | VARCHAR(20) |
| promotion\_id | VARCHAR(20) |
| **Constraint** | constraint pk\_reservation primary key(res\_id) |
| CONSTRAINT FK\_reservation01 FOREIGN KEY (rent\_id) REFERENCES rent (rent\_id) |
| CONSTRAINT FK\_reservation02 FOREIGN KEY ( promotion\_id) REFERENCES promotion  ( promotion\_id) |

Graphical user interface, application, Word

Description automatically generated

# **Challenges and Suggestions for improvements**

During the analysis and implementation of this project, there were some challenges that were encountered. When analyzing the scenario, there was difficulty in understanding the usage of “rural-route address”. After having discussion with our lecturer, we found out that there were some places with no regular address. Challenges were also faced during the creation of the conceptual design. We applied some useful resources to help us understand what to do and how to do it.

The Car2Go database design was a new and unique experience for us with valuable results that will be beneficial and useful for our career in the future.

# **Conclusions and Future Work**

In this project, we learned how to design a database system for car rental companies that focus on database creation, good ER schema building rules, how to create a relational schema with the ER diagram, derive function dependencies, how to normalize a relational schema and how different entities interact with each other. We also learned how to design a system from a database perspective and how to manage, store, maintain and manipulate data accurately.

**Future Enhancements**

In the near future, we will make some additions to the system as follows:

--Providing customer-member relations to give extra discount to the members

--Providing payment methods:

* Checks.
* Debit cards.
* Credit cards.
* Mobile payments.
* Electronic bank transfers

--Providing insurance coverage

# **References**

“Database design tutorial: Learn data modeling.” [Online]. Avalables: <https://www.guru99.com/database-design.html>

<https://www.w3schools.com/sql/default.asp>

[SQL Tutorial (tutorialspoint.com)](https://www.tutorialspoint.com/sql/)

<https://www.geeksforgeeks.org/>

[1NF, 2NF, 3NF and BCNF in Database Normalization | Studytonight](https://www.studytonight.com/dbms/database-normalization.php)

[What are the Microsoft SQL database functions? - SQL Server | Microsoft Docs](https://docs.microsoft.com/en-us/sql/t-sql/functions/functions?view=sql-server-ver15)

<https://docs.microsoft.com/en-us/sql/t-sql/>