DMD Project – 3rd delivery – Fall 2017

Team from BS2-4 group – Ruslan Kim, Alexander Kim, Maksim Surkov, Denis Chernikov

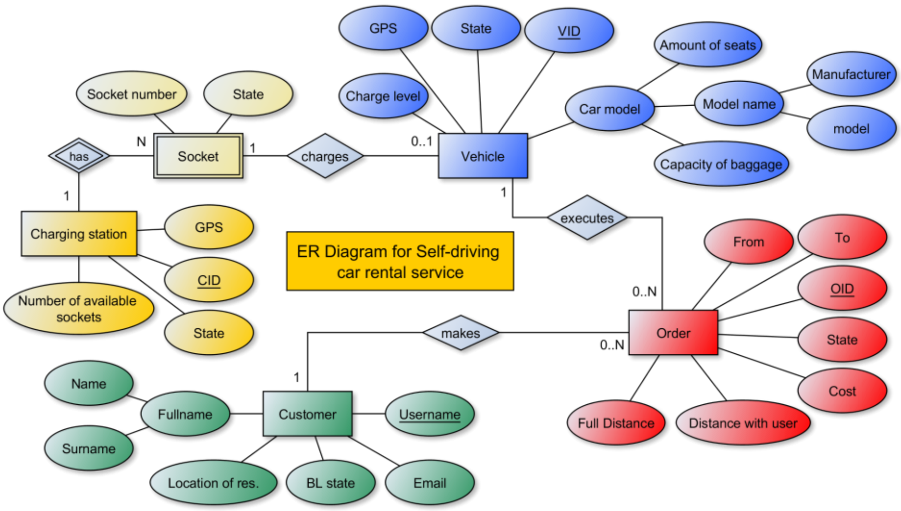
About delivery

We've chosen 1st option – database creation and SELECT queries implementation from the given list. DBMS is PostgreSQL. Our queries are:

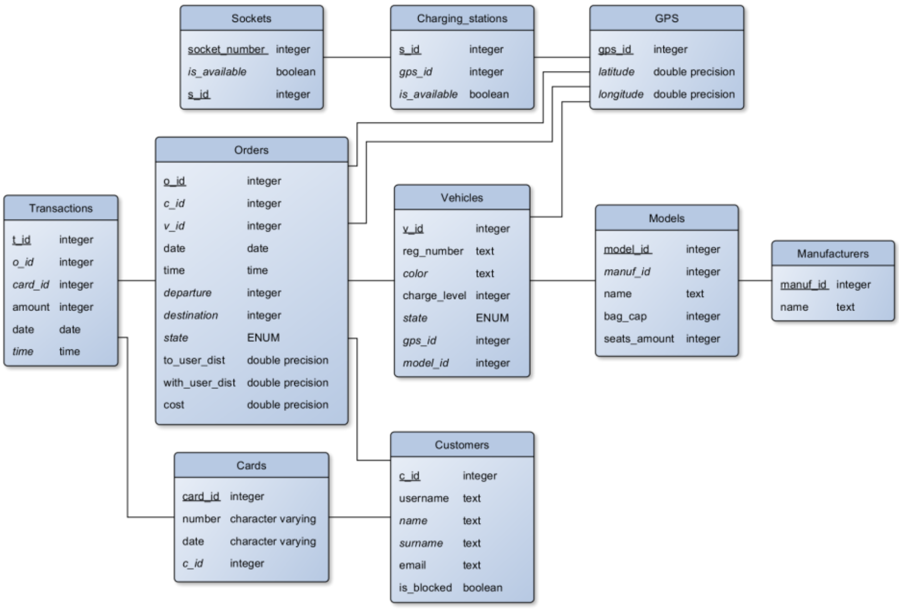
- 1 – Find all possible cars that match the description: car was red and its plate starts with "AN".
- 3 – Get statistics for one week on how many taxis are busy (% to the total amount of taxis) during the morning (7AM - 10 AM), afternoon (12AM - 2PM) and evening (5PM - 7PM) time.
- 4 – Check all payments for the last month to be sure that nothing was doubled (not specific user because hard to deal with random data, easily fixing by adding of one condition into SELECT).
- 5 – Get for some date statistics: average distance a car has to travel per day to customer's order location and average trip duration.
- 6 – Compute top-3 most popular pick-up locations and travel destination for each time of day: morning (7am-10am), afternoon (12am-2pm) and evening (5pm-7pm).

Changes in ER

Here is an ER diagram from the 2nd delivery:



And here is shorthand schema of the final database schema:



Sockets	
socket_number	integer
is_available	boolean
s_id	integer

Charging_stations	
s_id	integer
gps_id	integer
is_available	boolean

GPS	
gps_id	integer
latitude	double precision
longitude	double precision

Orders	
o_id	integer
c_id	integer
v_id	integer
date	date
time	time
departure	integer
destination	integer
state	ENUM
to_user_dist	double precision
with_user_dist	double precision
cost	double precision

Transactions	
t_id	integer
o_id	integer
card_id	integer
amount	integer
date	date
time	time

Vehicles	
v_id	integer
reg_number	text
color	text
charge_level	integer
state	ENUM
gps_id	integer
model_id	integer

Models	
model_id	integer
manuf_id	integer
name	text
bag_cap	integer
seats_amount	integer

Manufacturers	
manuf_id	integer
name	text

Cards	
card_id	integer
number	character varying
date	character varying
c_id	integer

Customers	
c_id	integer
username	text
name	text
surname	text
email	text
is_blocked	boolean

Let's compare and contrast two schemas: first version (ER-diagram) that was created during initial design and presented on previous assignments and second version which was directly extracted from PostgreSQL database.

Names of actual relations were renamed to be consistent (relations in plural form capitalized, attributes singular lowercase). Relations between entities from ER-diagram were fixed: e.g. Vehicle -> Model (one-to-many).

All GPS attributes were replaced with gps_id from new table "GPS" that stores coordinates and provides unified storage for GPS info. That helps to reduce access time which happen frequently in case of GPS.

Table Charging_stations: attribute "number of av. sockets" was removed, because this information can be achieved using query more easily.

Table "Sockets" is implemented as in ER-diagram.

Table "Vehicles" is implemented as in ER-diagram with addition of "color" and "reg_number" attributes.

Composite attribute "Car Model" was created as separate tables "Models" and "Manufacturer".

Table "Orders" was recreated in respect of foreign keys to relations "Customers" and "Vehicles".

New table "Transactions" was added with a dependency on "Orders" and "Cards".

Relation "Customers" is recreated as in ER-model plus augmentation with "Cards" table to keep records of customers' payment methods. "Transactions" also depends on "Cards."

How to check our assignment

First of all, you need to install “PostgreSQL” version 10.1. You can download it by the link to be consistent in used software (NOTE: you are very advised to check this solution using OS Windows).

When installed, set all the default data and remember manager’s password (you will need it to be able to make queries and commands, simply use “1” for example).

You can have a look at the dump file “`db_dump.psql`”, but it is pretty hard to determine data in it, and standard means of restoring the dump are hard to deal with... That’s why we advise you to use our generator of random data!

To set up the random data, use batch file “`INIT.BAT`” or launch Python script “`__main__.py`”. It will use Python scripts to create the database (using default user “postgres” and it’s password that you’ve set up while installed PostgreSQL). All the data will be generated randomly. You can do generation as many times as you want, it will create another set of random data in the same database.

To work with queries, use batch “`SELECTS.BAT`” or launch “`selects.py`”. Interface is clear and easy to deal with.

About any troubles of launching write to one of us.

Conclusion

This project gave us a lot of understanding of how to create and deal with efficient databases, and how hard is to start work with some new DBMS, but as more knowledge of it we get – then better is our work.

P.S.: It works on our machines :D