Data Modeling and Databases assignment #3

Github link: Link

Members:

Temur Kholmatov

Abdurasul Rakhimov

Rishat Maksudov

Group: BS17-05

General Information:

The required version of Python is 3.6 or later. Other requirements are presented in file "requirements.txt" and can be installed using pip. At the very beginning, you must run "main_local.py" file. There you need to fill username and password, and also choose the first action:

To create a database with tables (need to enter "1")

To load from backup (need to enter "2") - recommended

Just to run GUI (need to enter "3")

To create a sample database only (need to enter "4")

Creation of tables is located in "create_table.py" file and filling database with some data is located in folder "sample_data". GUI creation is located in folder "gui_application".

For GUI we use ***Tkinder*** and ***Pandastable*** libraries.

If you have some troubles, contact us via Telegram: @temur_kholmatov

Queries explanation:

1. Simulated the scenario by using method ***preload_data***. By attributes, we find plates of the cars and then find the exact car from ***rent_records*** and customer at exact day. As result, we must get the table of cars (maybe empty table if the car does not exist).
2. Simulated the scenario by using method ***preload_data***. In order to show the number of occupied sockets for each hour, we select from ***charging_station_sockets*** with exact time and date. As result, we must get a table with hour column and count column.
3. No need to simulate the scenario. Count number of ***rent_records*** that were busy during each allotted time (morning, afternoon, evening), and divide by the number of all cars in the system. As result, we must get columns for each daytime.
4. Simulated the scenario by using method ***preload_data***. Check accordance between ***rent_records*** and ***payment_records*** of the exact customer. Group them by id and see ***no_of_transactions***. As result, we must get an empty table (because nothing was doubled).
5. ***Modified***. As we do not track distance a car has to travel to customer's location, we just deleted this point from the query. Simulate the scenario by using method ***preload_data***. Then we select all ***rent_records*** with the exact date and find its average trip duration (***date_to*** – ***date_from***). As result, we must get an average duration of the trip in minutes.
6. ***Modified***. As we do not track pick-up locations, we just show a table with ***rent_records*** according to allotted time (morning, afternoon, evening).
7. No need to simulate the scenario. Count number of cars, find limit = 10% of all cars. Then count the number of ***rent_records*** for each car during last month, group by plates and order by limit number.
8. Simulated the scenario by using method ***preload_data***. For each customer, we count the number of trips, when a car he used was charged. All dates were selected according to exact allotted time (month). There can be such a few information for output because ***sample_data*** take dates for ***rent_records*** and ***charge_records*** in the interim of year, that is why there are not so many intersecting dates of rents and charges for cars.
9. ***Modified***. As we do not track which ***car_parts*** were used in repair. No need to simulate the scenario. For each workshop, we choose the type of ***car_part*** that is mostly ordered and then show it in the format of (***workshop_id***, type, amount).
10. No need to simulate the scenario. We combine prices from ***repair_records*** and ***charging_records*** for each plate(car) and then show the only car with the highest average cost per day.