LINGI 2172: Databases
Mission 4

Quentin Gusbin
Nicolas Vrielynck

April 26, 2016

1 Data Model

We implemented the data model provided in the assignement. We changed some column names like "order" and "table" because those words are already used by the sql language. We used a serial id for every id in the model. We also added a new relation to the model. The relation "CLIENT_EMPLACEMENT" Which show what client is on which table currently. Thanks to this table we can can keep track of all the clients and drink ordered by those client.

For more information about the data model, check DataModel.sql

2 Core operations

In order to implement the 4 operations in the part 1 and 2, we added triggers. When we add an order, we first check that the client exist and is on a table thanks to the trigger and then add it if no error was found. The same strategy is used when adding drink, we check that the drink exist and that the quantity inserted is greater than 0. We used the last trigger for the payments. When a payment is registered, we check the amount and if it is greater or equal than the total the client has to pay, we delete the relation client_emplacement of the client from the database. This make the client unable to order and the table free again. Using such a structure allows us to keep an historic of the client, orders and ordered dink.

We implemented the 4 core operations with plpgsql function. The four function respect the in, pre, post and out given in the assignment. This is done thanks to the triggers and some special added check ad the beggining of each functions. We added a fifth function "TOTAL_AMOUNT" that we added for personnal use. When we check the amount paid by a user we just need to call our function to know how much he needs to pay.

For more information about the triggers and the core operations, check Trigger.sql and CoreOperation.sql

3 Scenario

We implemented the scenario in sql and python for the part 1 and 2. The scenario is exactly the same. We just print more informations with the python code since sql can not print multiple response.

First,we get the first free table and raise an exception if no such table was found. We then acquire the table and we get our client token. We order the sparkling water for this client by first looking for the id in the DRINK table. We then call the issue_ticket function and order another sparkling water. To finish the scenario we look up the total amount to pay with our function and pay it.

4 Higher-level database connectivity

For the higher level implementation we used SQLalchemy which is a python library for query building and ORM for most popular databases. For this part of the project we used the query building part of the library.

In order to reduce code redondoncy we created sub functions to check the pre conditions that where the same for most functions we had to implement.

After implementing the functions with SQL and in python we can see that the query builder has some advantages and disadvantages over the sql immplementation. Most programmer being more familiar with usual programming language like python or java will prefer to implement those rules and trigger in that language even thought it requiere additionnal sql requests. Another advantage is that it is more portable since not all databases procedure,trigger or rules. On the other hand, using those seems more secure since there shouldn't be any incoherence in the database when using those built-in database tools. It is easier to ensure database coherence if the rules are inside the database than if we have to check the requirements are met every time we acces the database.

In the end a query builder is not very felxible because we are very dependant on what function the library implemented. Sqlalchemy seems like a very complete library for query building but it seems always easier to understand sql than having to learn a query building library for every programming langauge we use. There isn't a right way to implement it. We defintaly shouldn't implement everything with triggers,rule and procedure as we shouldn't do everything in a higher level programming langauge either. The right way to do it lays somehwere in the middle but it is obviously different for every single project.