Bases de Dados

Lab 08 *Funções e Triggers*

This guide is the continuation of Lab 1. If you have completed the previous guides, you should have the database of the Bank example already been created in your database account on the server `db.ist.utl.pt`. If you need to create the Bank database again, please refer to the instructions in Lab 01.

***Tip**: Make sure you test all SQL statements, as well as the functions, and triggers appropriately. Use the psql command line console. If need, insert new values in tables and verify the results using the appropriate SQL queries.*

Stored Procedures and Functions

Implement and test the following functionalities:

(a) Write a function in **pgplsql** that returns absolute net worth of a client, that is, the difference between (1) all the money that customer has in accounts and (2) all the amounts he owes in loans to the bank. This function should have a parameter that identifies the customer (whose net worth is to be obtained).

(b) Write a function that returns the difference between the average balance of all accounts between two given branches. The function must have two parameters that identify the branches to compare.

(c) Using the function developed in the previous paragraph, write an SQL query that determines which is the branch whose average account balance is the highest among all agencies.

(d) The depositor table associates customers with accounts. Type the SQL commands necessary to alter the **depositor** table, so that if an account is deleted from the **account** table, the records in the **depositor** table that refer to that account are also deleted (**Tip**: Use ALTER TABLE and ON DELETE CASCADE) .

(e) Repeat the previous paragraph for the case of loans and the corresponding tables e **borrower** and **loan**.

Triggers

Implement and test the following *triggers*:

(a) Write a *trigger* that eliminates a loan from a customer when the respective amount owed is less than, or equal to zero. Consider that the down payments of a loan are made by updating the amount field in the **loan** table (executing the UPDATE instruction). The trigger must also ensure that when the amount is negative, the (negative) value is added as an asset of the respective branch, and the respective loan is eliminated from the borrower and loan tables. For example, if the amount owed is 100 and an update of 150 is made, the loan must be eliminated, and the branch assets must increase by 50.

(b) Suppose that the bank has decided that it will stop providing loans to customers who are not associated with accounts. Write a *trigger* that prevents this case, i.e., the customer from associating with the loan.