Heinrich Heine
University Düsseldorf

Institut für Informatik
Datenbanken und Informationssysteme
Prof. Dr. Stefan Conrad
Thomas Germer

WiSe 24/25
Assignment 3
Due November 5, 8:00 AM

Relational Databases and Data Analysis

- As usual, use `zipme.py` to create the archive to upload in ILIAS.

Test Environment

The tests of the following exercise will make use of a PostgreSQL database and the `psycopg` library to connect to the database. You will have to set the following environment variables to enable clients (e.g. `psycopg` or `psql`) to establish a connection to the PostgreSQL database.

- PGHOSTADDR – the IP address of the PostgreSQL server.
- PGUSER – the user name. It should be `postgres`.
- PGPASSWORD – the password of the PostgreSQL user chosen by yourself.

The following commands will start a database and set the relevant environment variables. You might recognize most of it from assignment 0.

- Start a Docker container named "mydb" running a PostgreSQL server. Make sure to replace `chooseGoodPassword` with a good password.

```
docker run \
    --name mydb \
    --rm \
    -p 5432:5432 \
    -e POSTGRES_PASSWORD="chooseGoodPassword" \
    postgres:17.0
```

- Set the user name.

```
export PGUSER=postgres
```

- Set your password. Later, when we test your solution, we will start our own server and set our own environment variables.

```
export PGPASSWORD="chooseGoodPassword"
```

- Get the IP address of the docker container.

```
docker inspect mydb | grep IPAddress
```

- Set the IP address from the previous step (172.17.0.2 in my case, but might be different for you).

```
export PGHOST='172.17.0.2'
```

  You should now be able to run the tests and connect to the database with the command `psql` if you have a PostgreSQL client installed.

Exercise 1 *SQL with PostgreSQL* (7 Points)

You are given the following relational model of a supermarket chain. The foreign keys in the relation `purchase` (overlined) reference the corresponding primary keys in the relations `product`, `store` and `customer` (underlined). The value `amount` of the relation `product` is greater than zero.

`product(product_id, price, productname)`

`customer(customer_id, firstname, lastname)`

`store(store_id)`

`purchase(purchase_id, customer_id, store_id, product_id, amount)`

Formulate the following queries using SQL in the corresponding Python files and run the tests, which will populate the database automatically. Make sure to start the `PostgreSQL` server and to set the environment variables so that `psycopg` can connect to the database beforehand. Do not hardcore any IDs, names or similar. The queries should also work if the database contains different products, customers, stores and purchases.

(a) Get all products (all attributes) with prices that are multiples of 10 cents (e.g. 3.50 €).

(b) Write a query which results in a greeting like `Hello, <firstname> <lastname>` for each customer whose last name contains the character "W" (case insensitive). The resulting column of the result should be called `greeting`.

(c) Get all products which have been purchased at least 20 times in total and how often they have been purchased in total (`product_id`, `total_amount`). Make use of the `amount` column of the purchase table. For example, when a `purchase` has an `amount` value of 5, the associated product has been purchased at least 5 times.

(d) Find the customer who has spent the most money on purchases as well as the total money spent by that customer (`customer_id`, `total_money_spent`). Consider the price of each product as well as the amount of each product purchased. If there are multiple customers who have spent the same amount of money, return any one of them.

(e) Find the customers who have purchased the fewest number of products as well as the total amount of products that customer purchased (`customer_id`, `total_amount`). Only consider customers who have purchased at least one product. If there are multiple customers who have purchased the same number of products, return all of them. For example, if there are two customers who in total have purchased two products each, return both of them.

(f) Some customers have the value NULL as their last name. Set their last name to "Smith". There is no need to select anything in this exercise. The test script will check whether the table has been modified correctly.

(g) Delete the customer with first name "Test" and last name "Customer". There is no need to select anything in this exercise. The test script will check whether the table has been modified correctly.

Exercise 2 *Multi-Dimensional Data Model* (3 Points)

(a) Facts can be categorized as additive, semi-additive or non-additive. Given an example of a fact for each category and explain why the fact belongs to that category. The file name of your answer must be `exercise_2_a.txt`.