Data Engineer Test

Data Wrangling

1. This exercise requires all necessary code to be written in any programming language (Preferably Python or Java). The goal is to retrieve the metric of municipal work related accidents provided by the data based on the Public Brazilian Social Security API (http://api.dataprev.gov.br/doc/index.html). As a first step, retrieve data for all years saving it in a .csv file and then transform and analyze the data to get the top 3 brazilian cities on the dataset, which grew the metric the most between 2003 and 2007, by geopolitical region.

Data Warehousing - Entity Relationship Diagram

2. Imagine that ContaAzul would be planning the introduction of a group feature between users of the same company. Please draw an entity relationship diagram including necessary tables, data fields, and relationships that supports the following use cases around our new groups feature:

   - Groups have specific characteristics such as a name, description, when they were created, what kind of financial activity they perform and other data fields you deem important.

   - New groups can be created only by the account admin on the mobile and webapp. It is important to know from which touch point (i.e. app) a group was created.

   - A creator of a group is able to invite other registered users to become administrator of the group.

   - Registered users can join different groups and become a member. In case a group is set to "private", the membership request has to be manually approved by the creator or an admin. Members of a group can then decide at any point in time to leave again.

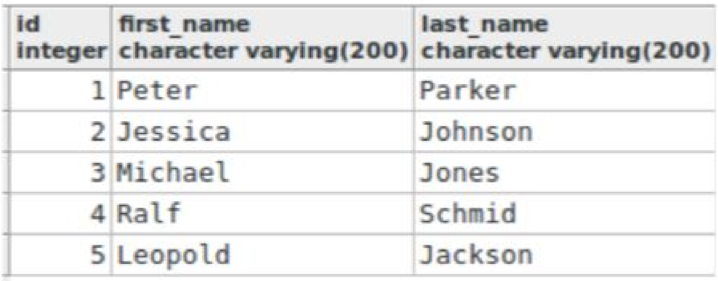
   - Groups allow features like creating process documentation, posting, commenting and liking posts or comments .

   Feel free to add other use cases that would enhance such a group feature at ContaAzul!

Data Exploration (postgreSQL)

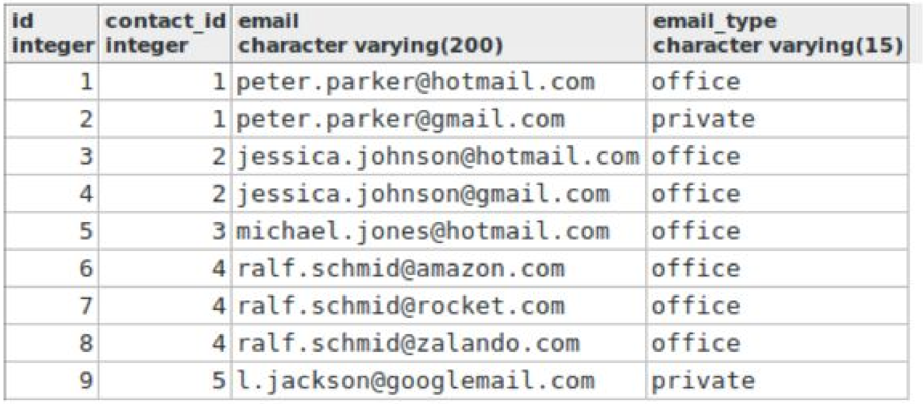
3. Using postgreSQL, with a list of client contacts is provided in table 1 with additional email information in table 2. We identified the problem that we have stored two or more email addresses of the same email type for a single contact e.g. as represented by the example of Jessica Johnson with 2 office email addresses. Create a query that outputs all contacts together with exactly 1 office email address whereby private email addresses should be excluded. We assume that the highest email id is the latest and most accurate record. In the Appendix, you will have the statements to create tables with sample rows.

Table 1: Contacts



id integer	first_name character varying(200)	last_name character varying(200)
1	Peter	Parker
2	Jessica	Johnson
3	Michael	Jones
4	Ralf	Schmid
5	Leopold	Jackson

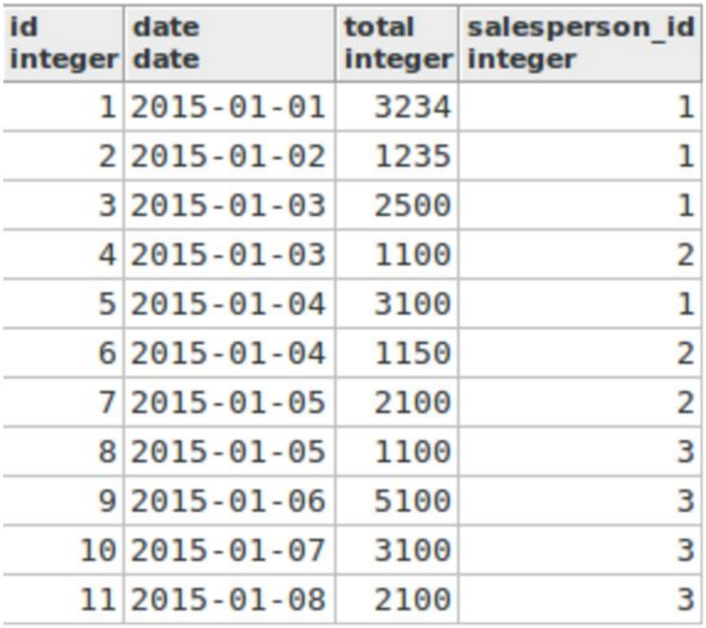
Table 2: Emails



id integer	contact_id integer	email character varying(200)	email_type character varying(15)
1	1	peter.parker@hotmail.com	office
2	1	peter.parker@gmail.com	private
3	2	jessica.johnson@hotmail.com	office
4	2	jessica.johnson@gmail.com	office
5	3	michael.jones@hotmail.com	office
6	4	ralf.schmid@amazon.com	office
7	4	ralf.schmid@rocket.com	office
8	4	ralf.schmid@zalando.com	office
9	5	l.jackson@googlemail.com	private

4. Using postgreSQL, the Table 1 contains transactions including the transaction ID, the date of the transaction, the total transaction amount in R$ as well as the ID of the respective sales person. The goal is to create a query that only returns the penultimate transaction of each salesperson. In the Appendix, you will have the statements to create the table with sample rows.

Table 1: Transactions



id integer	date date	total integer	salesperson_id integer
1	2015-01-01	3234	1
2	2015-01-02	1235	1
3	2015-01-03	2500	1
4	2015-01-03	1100	2
5	2015-01-04	3100	1
6	2015-01-04	1150	2
7	2015-01-05	2100	2
8	2015-01-05	1100	3
9	2015-01-06	5100	3
10	2015-01-07	3100	3
11	2015-01-08	2100	3

SQL Appendix

Data Exploration - Question 3

```
/* (Postgre SQL) SQL Query 1: Two Office Email Addreses - One Person */

/*Supporting tables if needed */
CREATE TABLE contacts( id int, first_name character varying(200), last_name character varying(200) );
INSERT INTO contacts (id, first_name, last_name) Values('1','Peter','Parker');
INSERT INTO contacts (id, first_name, last_name) Values('2','Jessica','Johnson');
INSERT INTO contacts (id, first_name, last_name) Values('3','Michael','Jones');
INSERT INTO contacts (id, first_name, last_name) Values('4','Ralf','Schmid');
INSERT INTO contacts (id, first_name, last_name) Values('5','Leopold','Jackson');

CREATE TABLE emails ( id int, contact_id int, email character varying(200), email_type character varying(15) );
INSERT INTO emails (id, contact_id, email, email_type) Values('1', '1', 'peter.parker@hotmail.com', 'office');
INSERT INTO emails (id, contact_id, email, email_type) Values('2', '1', 'peter.parker@gmail.com', 'private');
INSERT INTO emails (id, contact_id, email, email_type) Values('3', '2', 'jessica.johnson@hotmail.com', 'office');
INSERT INTO emails (id, contact_id, email, email_type) Values('4', '2', 'jessica.johnson@gmail.com', 'office');
INSERT INTO emails (id, contact_id, email, email_type) Values('5', '3', 'michael.jones@hotmail.com', 'office');
INSERT INTO emails (id, contact_id, email, email_type) Values('6', '4', 'ralf.schmid@amazon.com', 'office');
INSERT INTO emails (id, contact_id, email, email_type) Values('7', '4', 'ralf.schmid@rocket.com', 'office');
INSERT INTO emails (id, contact_id, email, email_type) Values('8', '4', 'ralf.schmid@zalando.com', 'office');
INSERT INTO emails (id, contact_id, email, email_type) Values('9', '5', 'l.jackson@googlemail.com', 'private');
```

Data Exploration - Question 4

```
/* (Postgre SQL) SQL Query 2: Two Latest Transactions by Salesperson */

/*Supporting tables if needed */
CREATE TABLE transactions( id int, date date, total int, salesperson_id int );
INSERT INTO transactions(id, date, total, salesperson_id) Values('1', '2015-01-01', '3234', '1');
INSERT INTO transactions(id, date, total, salesperson_id) Values('2', '2015-01-02', '1235', '1');
INSERT INTO transactions(id, date, total, salesperson_id) Values('3', '2015-01-03', '2500', '1');
INSERT INTO transactions(id, date, total, salesperson_id) Values('4', '2015-01-03', '1100', '2');
INSERT INTO transactions(id, date, total, salesperson_id) Values('5', '2015-01-04', '3100', '1');
INSERT INTO transactions(id, date, total, salesperson_id) Values('6', '2015-01-04', '1150', '2');
INSERT INTO transactions(id, date, total, salesperson_id) Values('7', '2015-01-05', '2100', '2');
INSERT INTO transactions(id, date, total, salesperson_id) Values('8', '2015-01-05', '1100', '3');
INSERT INTO transactions(id, date, total, salesperson_id) Values('9', '2015-01-06', '5100', '3');
INSERT INTO transactions(id, date, total, salesperson_id) Values('10', '2015-01-07', '3100', '3');
INSERT INTO transactions(id, date, total, salesperson_id) Values('11', '2015-01-08', '2100', '3');
```