GROUP 25

A. Followed optimization process:

Initially the creation and loading of our tables took extremely long. We quickly noticed that assigning foreign keys for all respective primary keys (referenced in the provided documentation) was way to expensive and thus we removed this. With our initial queriesafterloading we noticed that this was not necessary for all keys so we removed these unnecessary references. This was great for speed. Going down from +-1 hour to +- 2 minutes for creating and filling tables and the first queries. We did include all primary keys as specified to aid searching through the databases. Then work started on the q2queries. We wanted to work with materialized views but soon noticed that this was much slower than we expected so we used only normal views and limited the amount of views we used all together.

While at first we were using the provided TeamSQL for our local testing, as the server was unavailable until close to original, we were forced to switch SQL editor as TeamSQL was discontinued during the course of MS1. We choose PGadmin III which was much better for keeping track of performance and saving/sharing our queries.

In our optimization process we also made use of ANALYZE VERBOSE after the creation of the tables. The VERBOSE part of this function enables us to gain insights in the efficiency of our code. The ANALYZE function itself collects statistics on how the code performs and on the basis of this information creates the the most efficient plan of action. Queries ran faster when we implemented ANALYZE.

B. A description on the chosen optimizations

We create views in the *q2create.sql* file to make our code more consistent and readable. Although creating views do not directly increase the performance (i.e. running speed) the code, unless they are materialized, they help regarding consistency and readability. The reason we did not use materialized views, is the fact that the running time became too large. Only 3 minutes was allowed in the auxiliary structures, while every materialization took more than that duration. Views can be reused, which helps preventing repeating SQL code. It prevents different developers writing the same SQL code, probably in different ways, to reach the same result. This then also helps to increase the readability of the code. It can be compared as creating functions in other scripts like Python.

In addition to views, we used primary keys. The primary key forces/requires unique values for that column of the dataset. Where usage of primary keys have multiple benefits, the most important one is the fact that it makes it easier to find unique rows in a datasets. No sequential search through the dataset is then needed. This significantly increases performance.