**Report - template**

Assignment 2 - MySQL

**Group**: 51

**Students**: Henrik Borge, Torbjørn Grande, Sondre Rogde

**Introduction**

Briefly explain the task and the problems you have solved. How did you work as a group? If you used Git, a link to the repository would be nice.

The task was to implement a structure for storing data on activities. Each activity was related to a user and multiple trackpoints. The trackpoints contains a timestamp with the coordinates of the user’s position along with altitude. This was done keeping in mind that we were storing data for an application similar to Strava (more on this under Discussion).

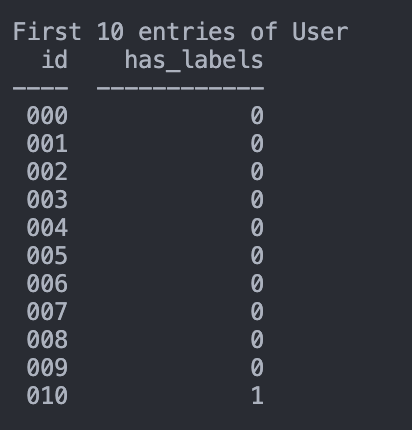
All team members knew each other well from before, so working as a group posed no problems. Being a team of three people really helped in discussing the technical details of how to store the data and what we had to consider. For most problems in part 2, we implemented a simple solution assuming the data was clean and made sure that solution worked. After implementing this simple solution, we expanded on it to deal with edge cases, invalid data and other things that could invalidate the output. This is discussed in greater detail under Results and Discussion.

We used Github for code collaboration and version control. The repository on Github can be found here: <https://github.com/Sondringsen/StoreDistribuerteOvinger>. The repository should be publicly available, but please let us know if there is something wrong with the access. The repository also contains a README.md containing all documentation required for running the code.

**Results**

Add your results from the tasks, both as text and screenshots. Short sentences are sufficient.

Small excerpts from the tables:



A screen shot of a black background

Description automatically generated

A screenshot of a computer

Description automatically generated

Question 1:

A black background with white text

Description automatically generated

Question 2:

A black background with white text

Description automatically generated

Question 3:

A screenshot of a cell phone

Description automatically generated

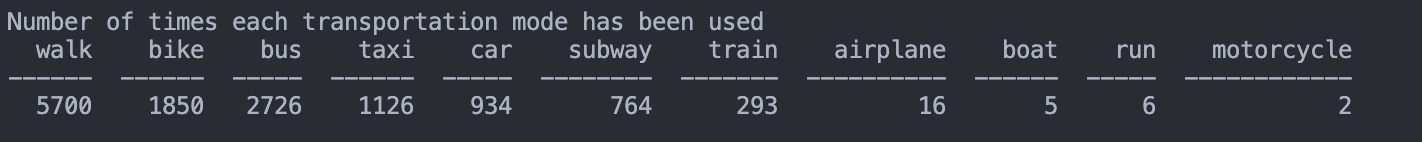
Question 4:

A screenshot of a computer

Description automatically generated

Question 5:

For this question our implementation worked very well since we only had to sum over all the binary variables in the Activity table.



Question 6a:

A screenshot of a computer screen

Description automatically generated

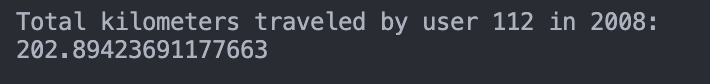
Question 6b:

A screenshot of a computer

Description automatically generated

Question 7:

For this question only 10km moves between two trackpoints were allowed. This was to avoid any faulty data where two consecutive trackpoints was too far from each other. For this specific user, it did not matter, however, it could matter for other users. Most of this task is done using pandas.



Question 8:

For this question only altitudes between -300 and 50,000 feet were allowed. All altitudes of -777 altitudes were dropped. Also, if the altitude changed with more than 300 feet it was discarded. For this question, the only terms in the sum are the terms where there was a positive difference in altitude between two trackpoints, i.e., where the user ascends. Most of this task is done using pandas.

A screenshot of a computer

Description automatically generated

Question 9:

We found that most users have invalid activities. This task was also done using some pandas.

A screenshot of a computer

Description automatically generated

Question 10:

For this question we allowed the latitude to be between 39.915 and 39.917 and the longitude to be between 116.396 and 116.398.

A screen shot of a computer

Description automatically generated

Question 11:

A screen shot of a computer program

Description automatically generated

**Discussion**

Discuss your solutions. Did you do anything differently than how it was explained in the assignment sheet, in that case why and how did that work? Were there any pain points or problems? What did you learn from this assignment?

What we did differently

We did some things a bit differently than what was described in the given description. They were only minor changes implemented to make the solution more realistic for an app similar to Strava. In Strava you can have multiple transportation modes per activity. You can for instance run and cycle in one activity in Strava. To better meet this requirement the table storing activities contained an column for each transportation mode stored as a BIT(). A different solution is to store a comma-separated list, but this is seen as an antipattern in relational databases. In addition, we stored the transportation mode for a given trackpoint as a string (VARCHAR). Trackpoints can only have one transportation mode since it is just a point in time and not an interval. Please see the figures below for a detailed description of the tables and datatypes.

A blue and white rectangular box with black text

Description automatically generated

A table with text on it

Description automatically generated

A table with text and numbers

Description automatically generated

Handling messy data

To make the functionality most similar to Strava we let any activity be defined by the .plt files with start time and end time equal to the min and max timestamp of that plt.file respectively. If an activity was labeled, but did not correspond to any TrackPoints, we added it as an activity, but flagged it as invalid. We did this because you can add activities in Strava without having any GPS tracking (i.e. no trackpoints). Before inserting the data, any trackpoints with highly unlikely values, for instance, the altitude was restricted to be between -300 and 30,000 feet were removed. The coordinates were also verified to be between -90 and 90 and -180 and 180 for latitude and longitude respectively. There is also some consecutive trackpoints which does not make sense, for instance, where the distance between them is unrealistically high. This is not handled when inserting the data but is handled during the queries where the distance between trackpoint-coordinates must be within a certain range. The same goes for altitude.

Pain points

Most of the exercise was completed without any great obstacles. The only pain point experienced was during part 2 with some of the more complicated queries. Those queries often required extracting large chunks of data and manipulating it in pandas. Since extracting and manipulating such large quantities of data is computationally intensive and time consuming it required careful attention to detail when coding to avoid spending to long debugging.

What we learned

You learn a lot of things doing projects like these. The most important one and relating to this course would be how to actually implement and maintain a database. Since we did not have access to a virtual machine from the start, we maintained a database locally hosted using Docker. Experience and knowledge about these things are hard to gain from a book and is best learned through doing projects. These skills are also essential for future employers. Additionally, we gained more experience with working with data in python and using pandas which is also highly appreciated by employers and very applicable to all sorts of problems be it academic or professional.

**Feedback**

Optional - give us feedback on the task if you have any. The assignment is new this semester and we would love to improve if there were any problems.

This was a very interesting exercise with the appropriate workload and difficulty.