Report - template

Assignment 3 - MongoDB

Group: 51
Students: Henrik Borge, Torbjørn Grande, Sondre Rogde

Note: to make grading easier, everything that is different from exercise 2 will be written in cyan. The screenshots are of course also different, but they are not marked in cyan.

Introduction

Briefly explain the task and the problems you have solved. How did you work as a group?

The task was to implement a structure for storing data on activities. Each activity was related to a user and potentially 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.

The following is small excerpts from the collections. We decided to implement the database in a slightly different way than the suggested structure to make the application a bit more flexible. This will be further discussed under Discussion.

First 10 entries of User

	_id	has_labels	activities
0	000	0	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14...
1	001	0	[156, 157, 158, 159, 160, 161, 162, 163, 164, ...
2	002	0	[213, 214, 215, 216, 217, 218, 219, 220, 221, ...
3	003	0	[359, 360, 361, 362, 363, 364, 365, 366, 367, ...
4	004	0	[620, 621, 622, 623, 624, 625, 626, 627, 628, ...
5	005	0	[966, 967, 968, 969, 970, 971, 972, 973, 974, ...
6	006	0	[1039, 1040, 1041, 1042, 1043, 1044, 1045, 104...
7	007	0	[1063, 1064, 1065, 1066, 1067, 1068, 1069, 107...
8	008	0	[1103, 1104, 1105, 1106, 1107, 1108, 1109, 111...
9	009	0	[1126, 1127, 1128, 1129, 1130, 1131, 1132, 113...

First 10 entries of Activity

	_id	user_id	walk	bike	bus	taxi	...	boat	run	motorcycle	valid_activity	start_date_time	end_date_time
0	1	000	0	0	0	0	...	0	0	0	1	2009:04:12 07:33:03	2009:04:12 17:24:59
1	2	000	0	0	0	0	...	0	0	0	1	2009:05:10 02:02:06	2009:05:10 06:20:11
2	3	000	0	0	0	0	...	0	0	0	1	2008:12:01 11:18:27	2008:12:01 11:35:15
3	4	000	0	0	0	0	...	0	0	0	1	2009:06:21 10:14:07	2009:06:21 14:47:12
4	5	000	0	0	0	0	...	0	0	0	1	2008:12:11 12:14:32	2008:12:11 12:24:32
5	6	000	0	0	0	0	...	0	0	0	1	2009:06:29 01:16:30	2009:06:29 11:13:12
6	7	000	0	0	0	0	...	0	0	0	1	2009:05:09 18:16:36	2009:05:09 18:30:36
7	8	000	0	0	0	0	...	0	0	0	1	2009:07:03 00:28:00	2009:07:03 08:37:23
8	9	000	0	0	0	0	...	0	0	0	1	2009:06:07 07:52:55	2009:06:07 09:46:03
9	10	000	0	0	0	0	...	0	0	0	1	2009:06:11 22:12:04	2009:06:11 22:34:54

First 10 entries of TrackPoint

	_id	activity_id	lat	lon	altitude	transportation_mode	date_days	date_time
0	671e5e0f302054856e5122c6	1	40.000017	116.327479	105	NaN	39915.314618	2009:04:12 07:33:03
1	671e5e0f302054856e5122c7	1	40.000168	116.327474	80	NaN	39915.314688	2009:04:12 07:33:09
2	671e5e0f302054856e5122c8	1	40.000055	116.327454	99	NaN	39915.314745	2009:04:12 07:33:14
3	671e5e0f302054856e5122c9	1	40.000021	116.327407	109	NaN	39915.314803	2009:04:12 07:33:19
4	671e5e0f302054856e5122ca	1	40.000035	116.327281	111	NaN	39915.314861	2009:04:12 07:33:24
5	671e5e0f302054856e5122cb	1	39.999983	116.327285	114	NaN	39915.314919	2009:04:12 07:33:29
6	671e5e0f302054856e5122cc	1	39.999853	116.327267	120	NaN	39915.314977	2009:04:12 07:33:34
7	671e5e0f302054856e5122cd	1	39.999745	116.327165	125	NaN	39915.315035	2009:04:12 07:33:39
8	671e5e0f302054856e5122ce	1	39.999661	116.326997	126	NaN	39915.315093	2009:04:12 07:33:44
9	671e5e0f302054856e5122cf	1	39.999528	116.326873	127	NaN	39915.315150	2009:04:12 07:33:49

Question 1:
Notice that there is a high number of activities. The reason for this will be discussed in the Discussion and has to do with our goal of making the application similar to Strava. It is clear that when including only activities where trackpoints are registered we get far fewer activities.

Including all activities:

```
Collection User has 182 documents
Collection Activity has 25004 documents
Collection TrackPoint has 9644128 documents
```

Including only activities with trackpoints:

```
Collection User has 182 documents
Collection Activity has 16048 documents
Collection TrackPoint has 9644128 documents
```

Question 2:
Average activities per user when including all activities:

```
Average activities per user:
144.53179190751445
```

Average activities per user when including only activities with trackpoints:

```
Average activities per user:
92.76300578034682
```

Question 3:
We see that 163 is on top here, but if we only include activities that has trackpoints, 128 would be on top.

Activity count for all activities:

```
Users with the most activities:
    _id  activity_count
0   163            3640
1   153            2294
2   128            2283
3   062            1046
4   085             949
5   167             854
6   068             853
7   025             715
8   144             569
9   075             509
10  126             450
11  052             404
12  041             399
13  084             391
14  004             346
15  140             345
16  010             335
17  112             308
18  147             291
19  017             265
```

Activity count for only activities which have trackpoints:

```
Users with the most activities:
    _id  activity_count
0   128            2102
1   153            1793
2   025             715
3   163             704
4   062             691
5   144             563
6   041             399
7   085             364
8   004             346
9   140             345
10  167             320
11  068             280
12  017             265
13  003             261
14  014             236
15  126             215
16  030             210
17  112             208
18  011             201
19  039             198
```

Question 4:
Including all activities:

```
Users who have taken taxi
      0
0   010
1   020
2   021
3   052
4   056
5   058
6   062
7   065
8   068
9   075
10  078
11  080
12  082
13  084
14  085
15  091
16  098
17  100
18  102
19  104
20  105
21  111
22  114
23  118
24  126
25  128
26  139
27  147
28  153
29  154
30  161
31  163
32  167
33  175
34  179
```

Only including activities which have trackpoint:

```
Users who have taken taxi
      0
0   010
1   021
2   052
3   056
4   058
5   062
6   065
7   078
8   080
9   084
10  085
11  098
12  102
13  105
14  111
15  114
16  126
17  128
18  139
19  153
20  161
21  163
22  167
23  175
```

Question 5:
For this question our implementation worked very well since we only had to sum over all the binary variables in the Activity collection.

All activities:

```
Number of times each transportation mode has been used
    _id  walk  bike   bus  taxi  car  subway  train  airplane  boat  run  motorcycle
0  None  5704  1850  2727  1126  934     764    293        16     5    6           2
```

Only including activities with trackpoints:

```
Number of times each transportation mode has been used
    _id  walk  bike  bus  taxi  car  subway  train  airplane  boat  run  motorcycle
0  None  1668   748  839   246  545     357     60         4     2    2           0
```

Question 6a:
In both question 6a and 6b we see that there is data registered for the year 2000. In the description of the dataset, it said that all activities were between 2007-2011, but

we have an activity from 2000 and 616 activities from 2012 which does not make sense. However, after taking a closer look at some of these activities, it did not seem like there was anything wrong with them and we decided to keep them. For instance, all the trackpoints to the activity from 2000 is shown below. It is only 3 trackpoints, but they all seem valid. Also, the numbers are quite high here and would be almost halfed if we did not include activities that has no trackpoints.

Total activities:

```
Number of activities per year
    _id  activity_count
0  2008           11229
1  2009            7732
2  2007            2021
3  2011            1872
4  2010            1511
5  2012             616
6  2000               1
```

Only including activities which have trackpoints:

```
Number of activities per year
    _id  activity_count
0  2008            5885
1  2009            5868
2  2010            1487
3  2011            1203
4  2007             994
5  2012             588
6  2000               1
```

Question 6b:
Depending on whether we include only the activities with trackpoints or all the activities the year with most time spent changes from 2008 to 2009.

Time spent on all activities:

```
Time spent on activities per year
    _id    time_spent
0  2008  15130.426667
1  2009  13564.297500
2  2007   4054.780000
3  2010   1717.335278
4  2011   1565.138611
5  2012    719.855833
6  2000      0.051111
```

Time spent on activities which have trackpoints:

```
Time spent on activities per year
    _id    time_spent
0  2009  11598.700000
1  2008   9180.187778
2  2007   2314.673333
3  2010   1388.036667
4  2011   1132.177778
5  2012    711.185833
6  2000      0.051111
```

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.

```
Total kilometers traveled by user 112 in 2008:
189.25483005938207
```

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.

```
Users who have gained the most meters
user_id
128    423543.767040
153    420085.574839
004    246928.843200
163    174346.023165
003    166500.657600
085    157892.739840
030    130048.406400
144    113923.985360
084    106138.065600
039    104687.827200
167     98996.059440
002     85703.054400
000     84559.749600
041     70973.990640
126     67962.467360
025     60714.575040
062     56677.712400
013     44247.511200
140     43994.562960
028     43863.463200
dtype: float64
```

Question 9:

We found that most users have invalid activities.

```
Invalid activities per user (only includes users who has invalid activities)
    user_id  invalid_activity_count
0       000                     101
1       001                      45
2       002                     100
3       003                     179
4       004                     219
..      ...                     ...
168     176                       8
169     178                       0
170     179                      28
171     180                       2
172     181                      14
```

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.

```
Users who have registered activities in the Forbidden City
   user_id
0      004
4      018
53     019
55     131
```

Question 11:
Including all activities:

```
Users most used transportation mode
user_id
010    walk
020    bike
021    walk
052     bus
053    walk
       ...
167    walk
170    walk
174     car
175     bus
179    walk
Length: 69, dtype: object
```

Only including activities with trackpoints:

```
Users most used transportation mode
user_id
010    taxi
020    bike
021    walk
052     bus
053    walk
       ...
167    bike
170    walk
174     car
175     bus
179    walk
Length: 69, dtype: object
```

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 a column for each transportation mode stored as a binary variable. 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. Trackpoints can only have one transportation mode since it is just a point in time and not an interval. This way we still have the ability to keep track of how much a user has walked and allow multiple transportation modes during one activity. We also decided to add a list of activity_ids to each user to increase performance of a few queries. We decided to not include data about the entire activity as this is a one-to-many relationship and not one-to-a-few. This was complemented by having a user_id on the many-side, that is each Activity document had a user_id field. Each TrackPoint had a field with activity_id, but the Activity documents did not have any reference to TrackPoint. This is because this is a one-to-quintillion relationship. This design scheme is in line with recommendation promoted by MongoDB (Zola, 2022).

To make the app more similar to Strava where you can register activities without tracking them (with trackpoints), we decided to let an activity be defined in two ways. Either, you have a plt file with trackpoints where all trackpoints is one activity (which can be either labeled or not). Or, you have a labeled activity without any trackpoints. This is also why the number of activities may seem inflated in some questions. If we did not allow this the number of activities would be much lower. In fact, it would only be 16048 as shown below.

INSERT SS

Please see the figures below for a detailed description of the tables and datatypes.

Column	Datatype
id	string
has_labels	bit
activities	list of activity_id

Column	Datatype
id	int
user_id	str
walk	bit
bike	bit
bus	bit
taxi	bit
car	bit
subway	bit
train	bit
airplane	bit
boat	bit
run	bit
motorcycle	bit
valid_activity	bit
start_date_time	datetime
end_date_time	datetime

Column	Datatype
id	int
activity_id	int
lat	double
lon	double
altitude	int
transportation_mode	string
date_days	double
date_time	datetime

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 were removed. For instance, the altitude was restricted to be between -300 and 50,000 feet. 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. Joins are quite easy with relational databases, but not so much with document-based ones. We solved this by retrieving documents and putting them into pandas dataframes which you can easily join. We also noticed that the performance of some of the queries which needed to join tables were better using MongoDB. However, this might be because when we used MongoDB the join was performed on client-side while the joins when using mysql were performed on server-side.

What we learned

We learned a lot doing this project especially since none of the group members had any prior experience with document-based databases. For this specific exercise the

schema design was very similar to that of the relational database design. However, it is easy to imagine other use cases where this would not be true, and the design of the document-based design would look very different from a relational-based design. After discussing and thinking about this in the group we think we are better able to determine a good design for a database and decide whether to use a document-based or a relational database.

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.

We really appreciated how this exercise extended exercise 2. It was interesting to see the differences between relational databases and document-based databases in a project like this. It makes comparison easier and makes you think more critically about what database type would be appropriate for other use cases.

References

Zola, W. (November 2, 2022). *6 Rules of Thumb for MongoDB Schema Design*. MongoDB. Retrieved: 28.10.2024 from:https://www.mongodb.com/blog/post/6-rules-of-thumb-for-mongodb-schema-design