Freeway data in MongoDB

CS588 / DATA CLOUD/CLUSTER MANAGEMENT

STUDENT NAME: MATTHEW BUI

PROFESSOR: DR. KRISTIN TUFTE

PORTLAND STATE UNIVERSITY

I. Query Demo

General information:

- data name: Freeway
- mongoDB cloud
- Programming Language: Python 3.7
- Connect mongoDB cloud using “pymongo” library.
- Use the “re” library for the regex expression.

I. Query Demo

1. Count high speeds: Find the number of speeds > 100 in the data set.

Execution plan:

- Straight forward.
- Use count_documents functions of MongoDB.
- Query directly from MongoDB.

```
count_high_speed_one_hour = loop.count_documents({'speed': {'$gt': 100}})
```

- Result:

```
Count high speed in data: 3855779
```

I. Query Demo

2. Volume: Find the total volume for the station Foster NB for Sept 21, 2011.

Execution plan:

- Query directly from MongoDB.
- Use regex search function of MongoDB for starttime pattern “2011-09-21”.
- Use aggregate function of MongoDB with match condition.

```
pat = re.compile(r'{}'.format(date),re.I)
condition = [{'locationtext':location},{'starttime':{'$regex':pat}}]
pipe = [{'$match': {'$and': condition}},{'$group': {'_id': None,'total': {'$sum': '$volume'}}}]
forster_sept_21_2011_volume = loop.aggregate(pipe)
```

- .Result:

The total volume in 2011-09-21 at Foster NB: 59124

I. Query Demo

3. Find travel time for station Foster NB for 5-minute intervals for Sept 22, 2011.

Execution plan:

- Query length of Foster NB only once from Mongo DB.
- Query all speed and starttime on Sep 22, 2011 at Foster NB directly from Mongo DB.
- Execute the python function to round starttime and put it into appropriate time intervals.
- For each time interval, calculate the average time travel

I. Query Demo

3. Find travel time for station Foster NB for 5-minute intervals for Sept 22, 2011.

Result:

Average travel time at 2011-09-22 00:00 is 96.69 second(s)
Average travel time at 2011-09-22 00:05 is 91.91 second(s)
Average travel time at 2011-09-22 00:10 is 97.4 second(s)

...

Average travel time at 2011-09-22 23:45 is 102.55 second(s)
Average travel time at 2011-09-22 23:50 is 90.51 second(s)
Average travel time at 2011-09-22 23:55 is 93.1 second(s)

I. Query Demo

4. Peak Period Travel Times: Find the average travel time for 7-9AM and 4-6PM on September 22, 2011 for the Foster NB. Report travel time in minutes.

Execution plan:

- Make a query average speed query for each time intervals.
- Make a query for the station length.
- In python program, for each time interval, calculate average time travel.
- Result:

Travel time for 7-9AM: 169.79 seconds

Travel time for 4-6PM: 113.55 seconds

II. Model evaluation

Good	Bad
- Avoid “JOIN” operation when querying for ‘locationtext’ and ‘highwayname’. - Save spaces for unnecessary duplication fields. - Supports for most question.	- ‘Starttime’ datatype is messy. - Increase size of ‘loop’and ’station’ collections. - ‘Length’ field is unnecessary in the ‘loop’ collection. - Hard to handle updates.

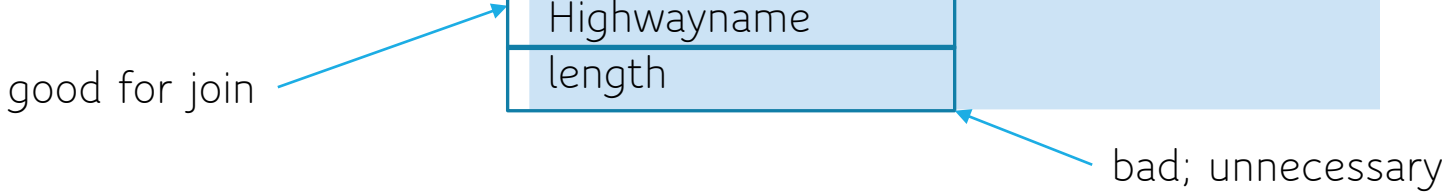
II. Model evaluation

Original

Loop
detectorid
Starttime
volume
Speed
Occupancy
Status
Dqflags

Modified

Loop
detectorid
Starttime
volume
Speed
Occupancy
Status
Dqflags
Locationtext
Highwayname
length



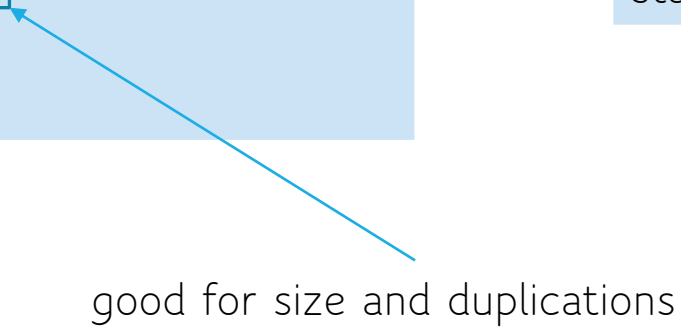
II. Model evaluation

Original

Loop
detectorid
milepost
locationtext
highwayid
detectorclass
numberlanes
stationid

Modified

detectors
detectorid
milepost
detectorclass
numberlanes
stationid



II. Model evaluation

detectors
detectorid
milepost
detectorclass
numberlanes
stationid

Loop
detectorid
Starttime
volume
Speed
Occupancy
Status
Dqflags
Locationtext
Highwayname
length

stations
Stationid
Highwayid
Milepost
Upstream
Downstream
Stationclass
Numberlanes
Latlon
Length
Highwayname
Locationtext

highways
Highwayid
Shortdirection
Direction
highwayname

Good: all information is reserved

Bad: duplication and data size

II. Model evaluation


```
{ "_id" : ObjectId("5ed8459a599c68c6491ced67"), "detectorid" : 1345, "starttime"
 : "2011-09-15 00:05:00-07", "volume" : 0, "speed" : "", "occupancy" : 0, "statu
s" : 0, "dqflags" : 0, "locationtext" : "Sunnyside NB", "highwayname" : "I-205",
 "length" : 0.94 }
{ "_id" : ObjectId("5ed8459a599c68c6491ced68"), "detectorid" : 1345, "starttime"
 : "2011-09-15 00:05:40-07", "volume" : 0, "speed" : "", "occupancy" : 0, "statu
s" : 0, "dqflags" : 0, "locationtext" : "Sunnyside NB", "highwayname" : "I-205",
 "length" : 0.94 }
{ "_id" : ObjectId("5ed8459a599c68c6491ced69"), "detectorid" : 1345, "starttime"
 : "2011-09-15 00:05:20-07", "volume" : 1, "speed" : 67, "occupancy" : 1, "statu
s" : 2, "dqflags" : 0, "locationtext" : "Sunnyside NB", "highwayname" : "I-205",
 "length" : 0.94 }
{ "_id" : ObjectId("5ed8459a599c68c6491ced6a"), "detectorid" : 1345, "starttime"
 : "2011-09-15 00:06:20-07", "volume" : 0, "speed" : "", "occupancy" : 0, "statu
s" : 0, "dqflags" : 0, "locationtext" : "Sunnyside NB", "highwayname" : "I-205",
```


Bad for query

II. Model evaluation

Future improvement:

- Investigate JOIN statistic deeply to avoid unnecessary fields. (e.g delete field from loop)
- Clean data starttime type with date type in mongodb.
- Avoid duplications by investigating dataset. (e.g highwayname)
- Use ‘partition’ on ‘starttime’ field.

III. Execution evaluation

Good	Bad
- Question 1 and 2 can be done query purely. - Question 3 only accesses to ‘loop’ once. - Question 4 gets main information from pure queries.	- Question 3 is slow because program must handle too much calculation. - Question 4 must access to the db twice.

III. Execution Evaluation

Question 3	Question 4
- Execution time: 5.3954689502716064, 3.6418819427490234, 4.315629720687866 - query access: 1	- Execution time: 4.488599061965942, 4.3257300853729 25, 4.244760036468506 - query access: 2

III. Execution Evaluation

- Question 3: move splitting operation to mongo query.
- Question 3: improve average speed queries for each time intervals.
- Question 3: handles string searching more cleverly.
- Question 4: merge two average speed queries accesses into one.

IV. What I have learn

Lessons – document dbs:

- Cleaning data is very important. (e.g starttime)
- Field duplicating is hard to avoid if we want to avoid join. (e.g locationtext)
- Duplications lead to consistency challenges. (e.g update, read)
- Duplications lead to size wasting. (locationtext)
- Merging all fields in one document increases the risk of wasting execution time. (e.g length)

Lessons – cloud managements:

- Investigate the data carefully to pick a right system. (e.g freeday not fit with document dbs)
- Investigate the data usage (questions) to design model better. (e.g join statistic)

IV. What I have learn

My advises for mongoDB and document DB users:

- document dbs fit well with the simple document data. (few column data)
- should take advantage of ‘partition’ on main attribute in mongoDB. (e.g: starttime)
- pymongo API is fully supported like original mongoDB API. (e.g: date converting function)
- mongoDB cloud uploading and connecting is a little bit challenging, so you should be prepared.

Appendix

Github link:

https://github.com/mbui0529/freewayDataCloud

Thank you