Part 2 - Data Modeling and Application Design

I. Data investigate:

The DBs must support these following questions (from Freeway data):

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

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

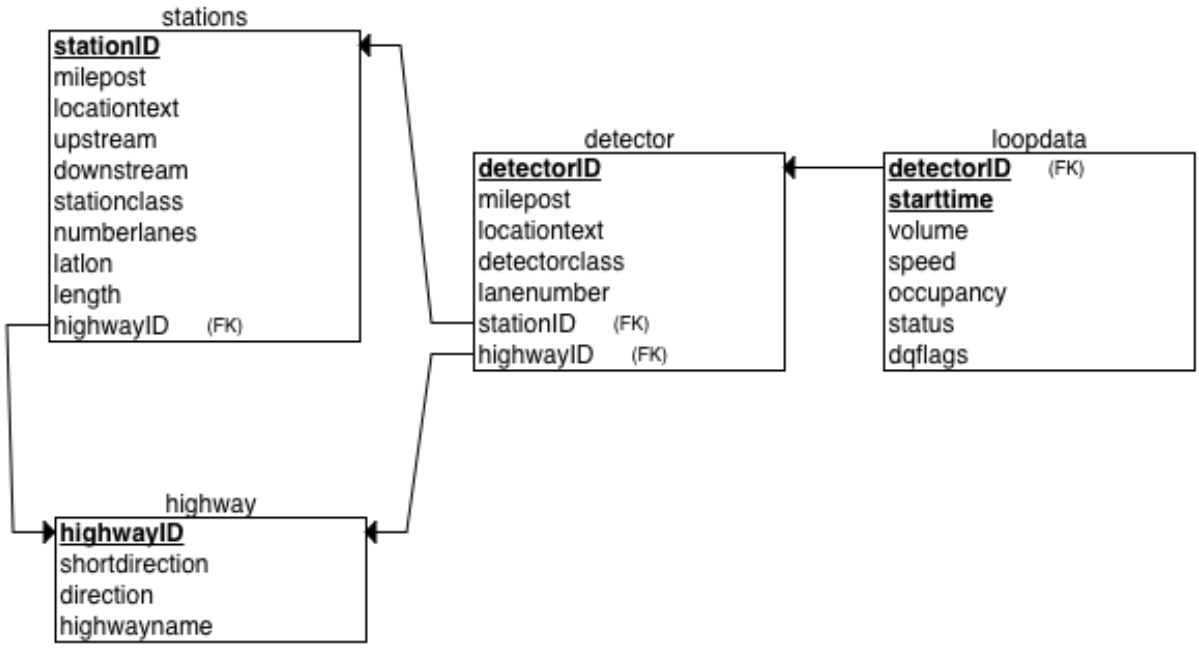
3) *Single-Day Station Travel Times:* Find travel time for station Foster NB for 5-minute intervals for Sept 22, 2011. Report travel time in seconds.

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

5) *Peak Period Travel Times:* Find the average travel time for 7-9AM and 4-6PM on September 22, 2011 for the I-205 NB freeway. Report travel time in minutes.

6) *Route Finding:* Find a route from Johnson Creek to Columbia Blvd on I-205 NB using the upstream and downstream fields.

The original RDBMS relational diagram:



II. Data Modeling & Database Design:

1. How you store the data structures in your data store?

1.1 Preferred design

The motivation: avoid JOIN by adding additional filed if needed.

Example loop data:
k, detectorID, starttime, volume, speed, occupancy, status, dqflags, locationtext, highwayname, length

Example detector data:
k, dectectorID , milepost, detectorclass, lanenumber, stationID

Example stations data:
k, stationID, milepost, locationtext, upstream, downstream, stationclass, numberlanes, latlon, length, highwayID, highwayname

Example highway data:
k, highwayID, shortdirection, direction, highwayname

indexes on: speed, starttime, locationtext, highwayname.

1.2 Variant design

The motivation: keep everything as the original RDBMS design and add more indexes to "JOIN" attributes and querying fields.

Example loop data:
k, detectorID, starttime, volume, speed, occupancy, status, dqflags.

Example detector data:
k, dectectorID, highwayID, milepost, locationtext, detectorclass, lanenumber, stationID

Example stations data:
k, stationID, milepost, locationtext, upstream, downstream, stationclass, numberlanes, latlon, length, highwayID

Example highway data:
k, highwayID, shortdirection, direction, highwayname

indexes on: detectorID, stationID, highwayID, speed, starttime, volume, locationtext, highwayname.

1.3 Explain why you believe your final design is the preferred one

My preferred design would save us from using "JOIN" operation, so it work faster. However, this have some cons such as it is hard to maintain and it takes time to modify the original data.

On the other hand, the variants does not require time for modify the original data. Also, it is easy to maintain and updates. However, we have to trade off the speed due to using "JOIN" as well as the consistency.

I choose mine because of the following reasons:

- The data for stations, highway, and detector is limited and fixed. It means we do not need to update them very often, so we can trade off the maintainability for speed.
- Due to my observation, we mostly query for loop data. As a result, the additional indexed field in loop data would make the DB run faster.

2. Describe the process you will use to restructure the data to fit your model.

- First, import the csv files to relational database using the relation prior schema.
- Second, use join queries and save the results to new tables.
- Lastly, import new csv files to mongoDB.
  mongoimport --host cluster0-shard-00-01-hvt3i.gcp.mongodb.net:27017 --ssl -u username -p 'password' --authenticationDatabase admin --db freeway --collection highways --drop --file filename.csv --type csv –headerline

3. Outline implementation strategies in pseudo-code based on the capabilities of your data store and the indexes you plan to define for all of the provided questions.

pseudo-code:

// *Count high speeds:* Find the number of speeds > 100 in the data set.
count = loop.count({ speed > 100 });

// *Volume*: Find the total volume for the station Foster NB for Sept 21, 2011.
volume = loop.sum({ { locationtext: 'Foster NB'}, {starttime: Sep 21, 2011},{ volume: 1 } });

// *Single-Day Station Travel Times:* Find travel time for station Foster NB for 5-minute intervals for Sept 22, 2011.

```
intervals = range(start = 0, end = 1440, step = 5)
result = []
length = db.findlength(locationtext = "Foster NB")
for each interval in intervals:
        speed = db.findspeed( starttime" in interval, locationtext = "Foster NB")
        result.append(speed / length)
```

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

```
intervals = [(7-9AM), (4-6PM)]
result = []
length = db.findlength(locationtext = “Foster NB”)
for each interval in intervals:
        speed = db.findspeed( starttime” in interval, locationtext = “Foster NB”)
        result.append(speed / length)
result = average(result)
```

// *Peak Period Travel Times:* Find the average travel time for 7-9AM and 4-6PM on September 22, 2011 for the I-205 NB freeway. Report travel time in minutes.

```
intervals = [(7-9AM), (4-6PM)]
result = []
length = db.findlength(highwayname = “I-205 NB”)
for each interval in intervals:
        speed = db.findspeed( starttime” in interval, locationtext = “Foster NB”)
        result.append(speed / length)
result = average(result)
```

// *Route Finding:* Find a route from Johnson Creek to Columbia Blvd on I-205 NB using the upstream and downstream fields.

```
map = db.findupstream&downstream (highwayname = “I-205 NB”)
mapArray = makeArray(map)
if (mapArray.find(“Johnson Cr NB”) & mapArray.find(“Columbia to I-205 NB”)):
        result = routefind(start = “Johnson Cr NB”, end = “Columbia Blvd”, map = map)

def routefind(start, end, map):
        if (start = end):
                return start
        else:
                index_start = map.findIndex(start)
                index_end = map.findIndex(end)
                return map[index_start:index_end]
```

4. Include proof that you have loaded at least a few data items into your system.

Show collection created:

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> show collections
detectors
highways
loop_one_hour
station
```

Show example data in highways collection

```
MongoDB Enterprise Cluster0-shard-0:PRIMARY> db.highways.find().pretty()
{
        "_id" : ObjectId("5ebe107493ec1ff5ea04a429"),
        "highwayid" : 3,
        "shortdirection" : "N",
        "direction" : "NORTH",
        "highwayname" : "I-205"
}
{
        "_id" : ObjectId("5ebe107493ec1ff5ea04a42a"),
        "highwayid" : 4,
        "shortdirection" : "S",
        "direction" : "SOUTH",
        "highwayname" : "I-205"
}
```

Show example data in stations collection:

```
{
        "_id" : ObjectId("5ebe17c21f216f3baec20271"),
        "stationid" : 1143,
        "highwayid" : 4,
        "milepost" : 21.12,
        "locationtext" : "Glisan to I-205 SB",
        "upstream" : 1141,
        "downstream" : 1051,
        "stationclass" : 1,
        "numberlanes" : 3,
        "latlon" : "45.5263291,-122.56548554",
        "length" : 1.53,
        "highwayname" : "I-205"
}
```

Show example data in detectors collection:

```
{
        "_id" : ObjectId("5ebe17a95d669b7cc4f89ed4"),
        "detectorid" : 1951,
        "milepost" : 21.12,
        "detectorclass" : 1,
        "lanenumber" : 3,
        "stationid" : 1142
}
{
        "_id" : ObjectId("5ebe17a95d669b7cc4f89ed5"),
        "detectorid" : 1941,
        "milepost" : 23.41,
        "detectorclass" : 1,
        "lanenumber" : 1,
        "stationid" : 1140
}
```

Show example data in loop collection:


```
{
        "_id" : ObjectId("5ebe1428133758bebe22d9e9"),
        "detectorid" : 1345,
        "starttime" : "9/15/11 0:06",
        "volume" : 0,
        "speed" : "",
        "occupancy" : 0,
        "status" : 0,
        "dqflags" : 0,
        "locationtext" : "Sunnyside NB",
        "highwayname" : "I-205",
        "length" : 0.94
}
```


5. Discuss how your design could handle updates to the data.

There are duplications of locationtext (in loop and station), highwayname (in loop, station, and highway), and length (in loop and station).

When adding the loop data, the new value must have the locationtext, highwayname, and length existed in target collections (stations and highways). If any of three fields is not founded in the target collections, the “adding operation” is not valid. This validation step can be done in programming part before inserting the new documents.

When updating the duplication value, we must update multiples collections. Because, like other no-SQL databases, MongoDB support single document transaction. However, we can still control consistency using read and write concern. The data is written very often because the detectors works continuously, so we set the DB to write optimization (set w: 1 and read: majority).