Homework Assignment #4

(Couchbase)

Your boss for your MongoDB assignment was quite satisfied with your work, but has a team full of developers who will be more comfortable with a language like SQL as opposed to MongoDB pipelines. You have been asked to continue working at this startup, but to move to Couchbase for your remaining tasks. Get a Couchbase Server "cluster" working, load the ShopALot data from the link provided in the setup document and run some queries to get more insights into the ShopALot data itself.

Get Ready… (and set)

To start, follow the setup instructions **carefully** in order to install and configure a local instance of Couchbase Server 6.6.2 on your machine. In addition to installation, the setup instructions detail how to create a bucket and load the ShopALot JSON data into your bucket. Note that for this assignment, everything will reside in ONE bucket. Couchbase 7.0 will offer a finer-grained collection-based storage model soon, but it's currently in Beta and we're going to stick with the latest non-Beta release. Buckets are kind of like databases, so Couchbase recommends that all documents for a given application reside in one collection (think mash-up table from the ER lecture). Consequently, each document must now be differentiated by an extra field (i.e. a new "type" field) to indicate its kind (e.g., orders, users, vehicles, …).

Go...!

It's time to get to work. Here's what you are to do using the Couchbase web UI:

1. To start, you should analyze the ShopALot schema for your entire bucket, similar to what you first did in your MongoDB assignment. Select "Query" (to navigate to the Query workbench for the Query service) on the left menu bar and expand "Data Insights". Hit the "Refresh" button in the lower right corner so that Couchbase will analyze a sample of the bucket. Doing so will help you answer the questions below:
    A. For the "**`stores`**" documents, what is the data type for the field "`categories`"?
    B. For the "**`orders`**" documents, what is the data type for the field "`time_placed`"?
    C. For the "**`users`**" documents, what is the data type for the field "`phones`"?
    D. For the "**`vehicles`**" documents, what is the data type for the field "`year`"?
2. Having better knowledge of the data's structure, you now want to run some queries to explore your documents further. Use the "Query Editor" to enter your N1QL query text. Use the *[indicated]* type field (for Query) or the relevant datasets (for Analytics) to differentiate between the various kinds of documents and answer the questions below.
    A. Try the query below and see if you are able to successfully execute it.

```
SELECT COUNT(*) FROM ShopALot WHERE `type` = "orders";
```

You should find that this query does not run. Couchbase requires that an index exists on your bucket before running any queries - it wants all queries to be supported by indexes. From here, one would ideally create additional indexes tailored to the queries you expect to run. We are going to create a primary index first, as this will contain ALL of the keys associated with each document in our bucket and will support any query (albeit slowly):

```
CREATE PRIMARY INDEX ON ShopALot;
```

Rerun the failed query and report back the number of orders that you have.

B. It's time to delve deeper into the data. Find the name of products that are in the "Bread & Bakery" category and have a list price greater than 3, limiting the number of results to 10. ***Note: Write your query to return a list of strings, not a list of JSON objects.*** (You will recognize this and a few other queries from your MongoDB assignment.) Record the amount of time it took for your query to execute. (You should be unsatisfied with that execution time). Inline in the UI by "Query Results", navigate to "Advice" and then create one of the recommended indexes. Run your query again and report the back a) the query itself, b) the time it took to run the query w/o the index, c) the time it took the query to run w/ the index, d) the create index statement you used, and e) the results of your query.

C. We would like to identify customers who have placed large orders. Return the customer ids associated with orders with a total price higher than $600 and with more than 1 item in that order. Again, your query should be written to return a list of customer id strings, not a list of JSON objects.

D. Your boss wants to have a list of shoppers who can be contacted when a large number of orders comes in. You are thus asked to search for shoppers that have a home phone number on file and who also have a 'capacity' field. Return just their user id and name fields. Sort your results in ascending order of last name, and print only the first 5 documents. ***Note: The words "last" and "first" are N1QL reserved words, so you will need to escape them using backticks (`) when referencing them.***

E. As discussed in part B), you probably noticed that your queries up to this point have been running fairly slow. One solution is to create one or more indexes for each query, but this becomes troublesome for larger ad-hoc queries. This is where the Analytics portion of Couchbase Server shines. First, navigate to the "Analytics" tab on the left menu in the Couchbase UI. From there, enter the following commands in the Analytics workbench to create Analytics datasets (which are like MongoDB collections) based on the data your main bucket:

```
CREATE DATASET orders ON ShopALot WHERE `type` = "orders";
CREATE DATASET own ON ShopALot WHERE `type` = "own";
CREATE DATASET products ON ShopALot WHERE `type` =
"products";
CREATE DATASET stockedby ON ShopALot WHERE `type` =
"stockedby";
CREATE DATASET stores ON ShopALot WHERE `type` = "stores";
CREATE DATASET users ON ShopALot WHERE `type` = "users";
CREATE DATASET vehicles ON ShopALot WHERE `type` =
"vehicles";
CREATE DATASET workfor ON ShopALot WHERE `type` = "workfor";
CONNECT LINK Local;
```

Your data will now be flowing into the Analytics service of Couchbase! Wait until there are no more remaining mutations. (You can view data ingestion progress on the right hand side of the UI under "Datasets".)
Now (for the question itself): Rewrite queries C and D to work using your new Analytics datasets and then run them here in the Analytics workbench. There are some subtle differences that you'll find between N1QL for Query and N1QL for Analytics. Compare how long it takes for queries C and D to run using the Query service vs. the Analytics service.

F. *Continue the rest of this assignment using your **Datasets** in the Analytics service. Your queries from this point forward should not involve the bucket name and will not need to include any type predicates!* Similar to query C, we would like to get more information on customers who have placed large orders. Return the names and the email addresses of customers who have placed one or more orders that totaled more than $600, as well as the orders' associated order_id and total_price, ordered by the total_price in descending order.
G. Aggregated reports are easier to read and present than large amounts of raw collected data. Your boss wants to see -- for every store -- its total number of orders, its average order price, and its maximum order price. Print just the store_id, count of orders, average order price, and maximum order price. Sort your results in descending order of the order counts, and print only the first 10 documents.
H. Your boss is now aggregate-query hungry! Your boss now wants to know, for each product category, the total number of stores that are associated with this category. (Take a look at the UNNEST operator to help with this query.)
I. **[Extra credit]** Print the user id, full name, and order statistics for customers who've ordered from Mapco and Country Market but not from 7-Eleven. The order statistics should be the total number of orders placed, the price of their cheapest order, and the price of their most expensive order, considering all of the orders that they've placed from anywhere. Organize the resulting report in order of ascending last names, and only report on customers who have placed more than two orders and whose e-mail provider is aol.com.

What To Turn In

When you have finished your assignment, you should use Gradescope to turn in a PDF file from **the template** on the course website that lists all statements you ran to create your indexes and run your queries. Please follow the following steps in order to generate the file for submission.

1. Open the Google Doc Template file, Click [File] -> [Make a Copy] will create a copy of this template. You can edit it and once the editing is done, you can download it as a PDF file and submit it to Gradescope.
2. Open the file. At the very top, specify (again) your name, student ID, and the date.
3. Fill in the query and notes to each question. **Only include the things specified in the template**. *To minimize errors, the majority of this assignment should be performed through the Couchbase UI.*
4. Validate as much as you can before submitting.
5. Finally, submit the PDF file into Canvas!