Lab 4: MongoDB Aggregation Pipeline

**Due date:** Monday February 4, 11:59pm.

# Lab Assignment

## Assignment Preparation

This is an individual lab.

## Assignment

For this lab, you will develop a number of MongoDB aggregation pipelines that extract and organize information from the Iowa liquor store sales dataset (and various collections you have built off of this dataset). Your pipelines can use `$match, $project, $group, $bucket, $unwind, $lookup, $sort, $limit, $count, $facet, $addFields` operations. There should be no need to use other commands (although variants using other operations are possible).

**Question 1.** This is a version of the `leaderboard.java` assignment from Lab 1. We need to find the vendor (or vendors - if there are multiple vendors tied) who sold the largest number of bottles to the stores in the state of Iowa. Write an aggregation pipeline that finds the vendor and reports the name and the Id of the vendor, and the total number of bottles sold in the following format:

```
{ vendorID: <vendor number>,
  vendor: <vendor name>,
  bottles: <number of bottles sold>}
```

**Collection.** Run the aggregation query on the `invoices` collection in the `iowa` database.

**Hint.** In the lab we did an exercise on how to find documents that contain information about the largest/smallest value. The difference between the exercise and this question is that in the exercise we already had the values (total sale price in dollars) available to us in the original documents. The question requires you to compute those values first.

**Question 2.** This is the version of the `countyStats.java` assignment from Lab 1. We want to find out the sales per capita for each county in Iowa. Write a MongoDB aggregation pipeline that for each county in Iowa reports the following information: the name of the county, the toal sales of liquor to stores in the county in dollars (sum of all `"Sale (Dollars)"` amounts), total volume of liquor in liters sold to stores in the county, and per capita sales in dollars and per capita volume in liters. The output shall be in the following format:

```
{ county: <CountyName>,
totalSales: <total sales in dollars>
totalVolume: <total volime in liters>
perCapSales: <per capita sales in dollars per person>
perCapVolume: <per capita volume in liters per person> }
```

Sort your output in descending order by per capita sales in dollars per person.

**Collection.** You can either run this aggregation pipeline on the `populations` collection of the `iowa` database, or on the `invoices` collection of the iowa database.

**Question 3.** We want to analyze the liquor distribution channels. For this purpose we want to create a bipartite graph connecting vendors (one set of nodes in the graph) to stores (the other set of nodes). Each node in such a graph if a pair `<vendor, store>` such that at some point in time, a vendor sold some alcohol to the store. We want to enhance the edges with some additional information. Specifically, we want to record the total number of sales (i.e., the number of distinct invoices) from the vendor to the store, the total number of bottles sold by the vendor to the store, and the total amount of money (in dollars) that was paid for the alcohol. Write a MongoDB aggregation pipeline that constructs such a bipartite graph as a collection of objects of the form:

```
{ vendor: { vid: <vendorId>,
            name: <vendor Name>},
  store: {  sid: <store number>,
            name: <store name>,
            County: <county>},
  NPurchases: <number of purchases>,
  BottlesSold: <number of bottles sold>,
  TotalAmount: <total amount of all sales in dollars>}
```

**Collection.** Run this aggregation pipeline on the `invoices` collection of the `iowa` database.

**Hint.** What is the "main" operation that makes the output possible?

**Question 4.** For each county find its favorite, in terms of total number of bottles sold to the stores in the county, category of liquor. Report the output in the form:

```
{County: <County name>,
 Category: <Category Name>,
 Bottles: <bottles sold to stores in county>}
```

**Collection.** Run this aggregation pipeline on the `invoices` collection of the `iowa` database.

**Hint.** This, again, is an extension of our lab excercise – we want to find what the largest number of bottles sold was in each county, and then use this information to find the appropriate categories.

**Question 5.** In the Iowa liquor sales dataset, the `"Sale (Dollars)"` key (attribute) contains the information about the total cost of the purchased alcohol at the `"State Bottle Retail"` price. That is:

$$\texttt{"State Bottle Retail"} * \texttt{"Bottles Sold"} = \texttt{"Sale (Dollars)"}$$

We also have information about the state bottle cost (`"State Bottle Cost"`). The difference between `"State Bottle Retail"` and `"State Bottle Cost"` is the profit per bottle of liquor.

Write a MongoDB aggregation pipeline that adds to the documents in the `invoice` collection three more keys: `"State Cost Total"`, `"Profit per Bottle"`, and `"Overall Profit"`. The keys store (not surprisingly), the state cost of all the bottles purchases (number of bottles times the state bottle cost), the profit per bottle in the current sale, and the overall profit from the sale.

**Collection.** Run this aggregation pipeline on the `invoices` collection of the `iowa` database.

**Question 6.** Now, let us find the most profitable vendor - i.e., the vendor (or vendors) that generated the largest amount of profit from its sales to the stores. We do this in two steps (this question and the next one).

Because different vendors have different numbers of sales and sell alcohol at different price points, the fair metric by which the vendors shall be judged

is the ratio of the overall profit made from the sale (`"Overall Profit"`) to the state cost total (`"State Cost Total"`) expressed as a percent (i.e., multipled by 100). For this question, write a MongoDB aggregation pipeline that for each vendor output the following information: vendor name, number of sales, and the average profit expressed as a percent as discussed above. The output format is:

```
{Vendor: <vendor name>,
 NSales: <total number of sales>,
 AvgProfit: <average profit from each sale>}
```

**Collection.** You can write this query to work with the `invoices` collection in the `iowa` database. Alternatively, you can create a collection `profits` in your own database that contains the solution to **Question 6**, and write the aggregation pipeline for it.

**Hints.** Two things. First, see the **Collection** note above. If you are writing a pipeline for the `invoices` collection - what do you need to do first? Second, be very careful with the order in which you aggregate. We want to compute the profit (as a percent) for each sale, and then take the average of those numbers. Make certian your pipeline reflects that.

**Question 7.** Now we are ready to actually find the most profitable vendor. Using the results of **Question 6** report the record (or records) of vendors who reaped the largest average profit from their sales. The records shall match the spect from **Question 6.**.

**Collection.** You can run this on either the `invoices` collection of the `iowa` database, or the `profits` collection in your database (see **Question 6**). Additionally, you can create `vendorProfits` collection in your database, put the output of **Question 6** in it, and create your aggregation pipeline for the `vendorProfits` collection[1].

**Hint.** The **Collection** part of this question is a hint enough. The remainder of the pipeline is in line with what we discussed during the lab exercises.

**Question 8.** Now let's figure it out in the other direction. Let's find the store (or stores) that paid the largest amounts in markups from the state bottle cost. We will do this, again, in two steps.

Thist time, though, our target is the total amount of money the store paid as markup. Note that the markup for the store is what the profit is for the

[1]If you choose to do it, simply duplicate the value of the `"Vendor"` key as the `"_id"` key in the collection.

vendor. So, first, for each store, let us collect the following information: (a) store name, (b) county the store resides in, (c) total number of purchases the store made, (d) total amount of money the store paid, (e) overall state cost of the purchased alcohol, (f) total the markup the store paid for the alcohol purchases. The format of the output is:

```
{ Store: <store name>,
  County: <county name>,
  NPurchases: <number of alcohol purchases>,
  TotalPaid: <total amount of money paid>,
  StateCost: <total state cost of all alcohol>,
  Markup: <total markup>
}
```

**Collection.** Either `invoices` in the `iowa` database, or `profits` (see **Question 6**) in your database.

**Hint.** Take appropriate care with your steps depending on which collections you are writing your pipeline for (if it is for the `invoices` collection, then what do you need to do first?). Otherwise, this should be similar to **Question 6** except possibly for the aggregation part.

**Question 9.** Now, let's find the store or stores (in case there is a tie) that paid the largest amount of money in total markup. Report the output in the same format as documents in `Question 8.`

**Collection.** Either `invoices` in the `iowa` database, or `profits` (see **Question 6**) in your database. Additionally, you can create a collection `storeMarkup` in your database, dump the results of **Question 8** in it, and write a pipeline for this collection.

**Note.** Yes, Questions 8 and 9 are repetitive w.r.t. Questions 6 and 7. This is done for a reason. I want you to remember the sequence of pipeline steps to get these type of operations by heart.

**Question 10.** We want to understand if there is a relationship in our dataset between the price of a single bottle of liquor purchased and the total number of bottles in the purchase. A typical way of quantifying relationships between two numeric variables is Pearson Correlation Coefficient.

Given two sets of observations $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ the Pearson correlation coefficient between $x$ and $y$ is computed as follows:

$$pearson(x, y) = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^{n}(x_i - \mu_x)^2)}\sqrt{\sum_{i=1}^{n}(y_i - \mu_y)^2}},$$

where $\mu_x = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\mu_y = \frac{1}{n}\sum_{i=1}^{n} y_i$ are the respective means of the two sets of observations $x$ and $y$.

The value of 1 means perfect positive correlation between the two sets of observations. The value of 0 means the two sets of observations are independent of each other. The value of $-1$ means perfect negative correlation between the two sets of observations (i.e., whenever one observation is significantly higher than its mean, the other observation is significantly lower).

In our case, we want to compute the Pearson correlation coefficient between the `"Bottles Sold"` and `"State Bottle Retail"` sets of values.

Write a MongoDB aggregation pipeline that computes the Pearson correlation coefficient between `"Bottles Sold"` and `"State Bottle Retail"` sets of values and returns it in the format:

```
{ pearson: <pearson correlation coefficient>}
```

(Your pipeline returns just one object with a single key in it).

**Collection.** Run this aggregation pipeline on the `invoices` collection of the `iowa` database.

**Hint.** There is a lot of "stuff" in the documents in the collection that you do not need. I recommend getting rid of it first. This may not be a necessary step, but it will help with debugging. Pearson correltation coefficient requires computing the mean of each of the sets of observations. You need this mean in your objects side by side with individual observation values. We have looked at a trick to do this - basically your goal is to compute the averages while preserving the original values used to compute them. Once you do this, the rest if arithmetics. There is a `$sqrt` operator that can be used to compute square roots.

**Question 11.** Now that you know how to compute Pearson correlation coefficient for the entire datasaet, let us see if the relationship between number of bottles purchased and the price of a bottle is different in different counties.

Write a MongoDB pipeline that computes the Pearson correlation coefficient between `"Bottles Sold"` and `"State Bottle Retail"` for each state, and returns the output sorted in descending order in the format:

```
{County: <county name>,
 pearson: <pearson correlation coefficient>}
```

**Collection.** Run this aggregation pipeline on the `invoices` collection of the `iowa` database.

## Submission

Place all your queries into a single file `lab04-queries.mongo`. Your name and email address shall be in the header comment (use Javascript comment syntax). Each query must have a comment identfying it ("Query 1" or "Question 1", at the very least). Use `README` file for any comments about your submission.

Use `handin` to submit as follows:

```
$ handin dekhtyar lab04 <FILES>
```

**Good Luck!**