Unit 9: Databases & The Modern Web:

Exercise 2 – Coffee Shop (Aggregation)

Assigned: 03/02/23

Due: N/A

Completed: 03/02/23

**LMS:** <https://lms.grandcircus.co/course/view.php?id=231&section=10>

**Google Doc:** <https://docs.google.com/document/d/1r4qZoL_lD_Xfl93UTy0EX9_8jEQi1JfGaQUP-67SWes/preview>

**GitHub:** TBD

**Task:** Write MongoDB commands using aggregation. Use pipeline steps: $sort, $limit, $skip, $match, $group, $project.

**Setup:** Use this data.

db.menu.insertMany([

 { "name": "Brewed Coffee", "price": 2.00, "type": "Coffee" },

 { "name": "Iced Coffee",   "price": 2.50, "type": "Coffee" },

 { "name": "Cafe au Lait",  "price": 2.25, "type": "Coffee" },

 { "name": "Espresso",      "price": 2.50, "type": "Espresso" },

 { "name": "Latte",         "price": 3.00, "type": "Espresso" },

 { "name": "Mocha Latte",   "price": 3.50, "type": "Espresso" },

 { "name": "Tea",           "price": 2.00, "type": "Tea" },

 { "name": "Chai",          "price": 3.50, "type": "Tea" },

 { "name": "Hot Chocolate", "price": 3.00, "type": "Other" }

]);

**Reference Slides:** <https://docs.google.com/presentation/d/e/2PACX-1vQOhDiSBYXzQOxPGBhFxZZ1lVXXWuGXmdOEb8jbHKNKGimXPFqLAZ3zuImDUHoX090Txs9ShEJr6C4G/pub?start=false&loop=false&delayms=3000&slide=id.gbf9072dbab_0_864>

**Build Specifications:** Write MongoDB shell commands using aggregation to produce the following reports. Record these commands in a JavaScript document and submit the document. You do not need to record the results of the commands. Expected results are shown below many questions.

1. List all drinks alphabetically by name. ($sort)
   * db.menu.aggregate([ { $sort: {name: 1} ])
2. List only the first 3 drinks, sorted alphabetically by name. ($sort, $limit)
   * db.menu.aggregate([ { $sort: { name: 1 } }, { $limit: 3} ])
3. List the next 3 drinks (4-6), sorted alphabetically by name. ($sort, $skip, $limit)
   * db.menu.aggregate([ { $sort: { name: 1 } }, { $skip: 3 }, { $limit: 3} ])
4. List all drinks alphabetically by name. Only show the name of each. ($sort, $project)
   * db.menu.aggregate( { $sort: { name: 1 } }, { $project: { item: ‘$name’ } } )
5. Get all drinks over 3 dollars. ($match)  
   { \_id: ..., name: "Mocha Latte", price: 3.5, type: "Espresso" }  
   { \_id: ..., name: "Chai", price: 3.5, type: "Tea" }
   * db.menu.aggregate([ { $match: { price: { $gt: 3 } } } ])
6. Get all espresso drinks.  
   { \_id: ..., name: "Espresso", price: 2.5, type: "Espresso" }  
   { \_id: ..., name: "Latte", price: 3, type: "Espresso" }  
   { \_id: ..., name: "Mocha Latte", price: 3.5, type: "Espresso" }
   * mb.menu.aggregate([ $match: { type: ‘Espresso’ } } ])
7. Get only the espresso type drinks. Sort them by price, highest first.  
   { \_id: ..., name : "Mocha Latte", price : 3.5, type : "Espresso" }

{ \_id: ..., name : "Latte", price : 3, type : "Espresso" }

{ \_id: ..., name : "Espresso", price : 2.5, type : "Espresso" }

* + db.menu.aggregate([ { $match: { type: "Espresso" } }, { $sort: { price: -1 } } ])

1. Get the lowest price of any drink. (Just the price.) ($group)  
   { \_id: null, price : 2.0 }
   * db.menu.aggregate([ { $group: { \_id: null, minPrice: { $min: ‘$price’ } } } ])
2. Get the average price of all drinks.  
   { \_id: null, price: 2.6944444444444446 }
   * db.menu.aggregate([ { $group: { \_id: null, averagePrice: { $avg: ‘$price’ } } } ])
3. Get the average price for each type of drink. Name it averagePrice.

{ \_id : "Tea", averagePrice : 2.75 }  
{ \_id : "Other", averagePrice : 3 }  
{ \_id : "Coffee", averagePrice : 2.25 }  
{ \_id : "Espresso", averagePrice : 3 }

* + db.menu.aggregate([ { $group: { \_id: ‘$type’, averagePrice: { $avg: ‘$price’ } } } ])

1. Get the most expensive price for each type of drink. Name it topPrice.

{ \_id : "Coffee", topPrice : 2.5 }  
{ \_id : "Espresso", topPrice : 3.5 }  
{ \_id : "Tea", topPrice : 3.5 }  
{ \_id : "Other", topPrice : 3 }

* + db.menu.aggregate([ { $group: { \_id: ‘$type’, topPrice: { $max: ‘$price’ } } } ])

1. Get the average price for each type of drink. The results should have field names “type” and “averagePrice”. ($project) Sort the results with the most expensive first.  
   { averagePrice : 3, type : "Espresso" }

{ averagePrice : 3, type : "Other" }

{ averagePrice : 2.75, type : "Tea" }

{ averagePrice : 2.25, type : "Coffee" }

* + db.menu.aggregate([ { } ])

1. Get the two most expensive drinks. ($sort, $limit)

{ \_id: ..., name : "Mocha Latte", price : 3.5, type : "Espresso" }  
{ \_id: ..., name : "Chai", price : 3.5, type : "Tea" }

* + db.menu.aggregate([ { $sort: { price: -1 } }, { $limit: 2} ])