Inconvenient Convenience Store

Milestone: Implementation in NoSQL

Group 21

Chandra Kiran Bestha

Kusuma Nara

617-238-4749

857-395-5608

Mail

bestha.c@northeastern.edu

nara.k@northeastern.edu

Percentage of Effort Contributed by Student1: 50%

Percentage of Effort Contributed by Student2: 50%

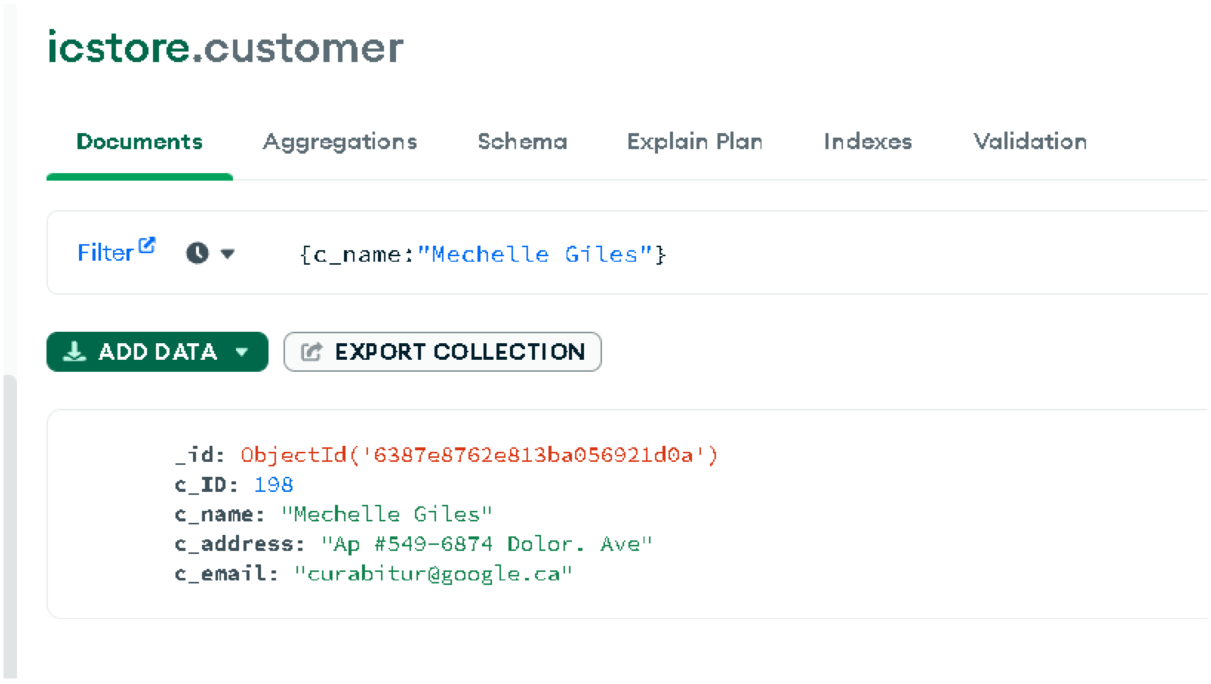
Signature of Student1: [illegible]

SignatureofStudent2: [illegible]

Submission Date: 11-30-2022

MongoDB has been used for the Implementation of database in NoSQL. MongoDB uses the MongoDB Query Language (MQL), designed for easy use by developers. Database of each class has been exported into JSON and then imported into MongoDB.

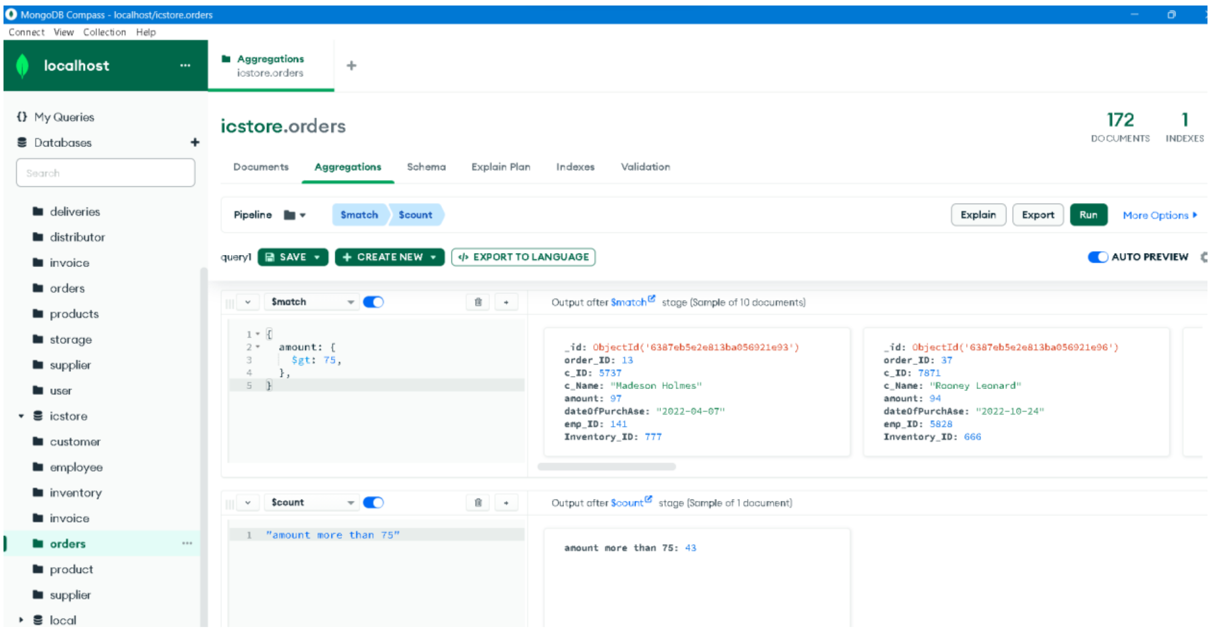
1) ***Find the customer named "Michelle Giles"?***



```
{c_name:"Mechelle Giles"}
```

```
_id: ObjectId('6387e8762e813ba056921d0a')
c_ID: 198
c_name: "Mechelle Giles"
c_address: "Ap #549-6874 Dolor. Ave"
c_email: "curabitur@google.ca"
```

2) ***Find the number of orders which valued above $75?***



$match

```
{
  amount: {
    $gt: 75,
  },
}
```

Output after $match stage (Sample of 10 documents)

```
_id: ObjectId('6387eb5e2e813ba056921e93')
order_ID: 13
c_ID: 5737
c_Name: "Madeson Holmes"
amount: 97
dateOfPurchAse: "2022-04-07"
emp_ID: 141
Inventory_ID: 777
```

```
_id: ObjectId('6387eb5e2e813ba056921e96')
order_ID: 37
c_ID: 7871
c_Name: "Rooney Leonard"
amount: 94
dateOfPurchAse: "2022-10-24"
emp_ID: 5828
Inventory_ID: 666
```

$count

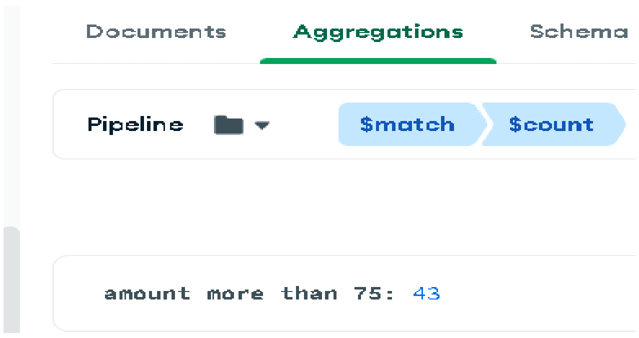
```
"amount more than 75"
```

Output after $count stage (Sample of 1 document)

```
amount more than 75: 43
```

The **$match** stage excludes orders that have a price of <= 75 to pass along the orders with price greater than 75 to the next stage.

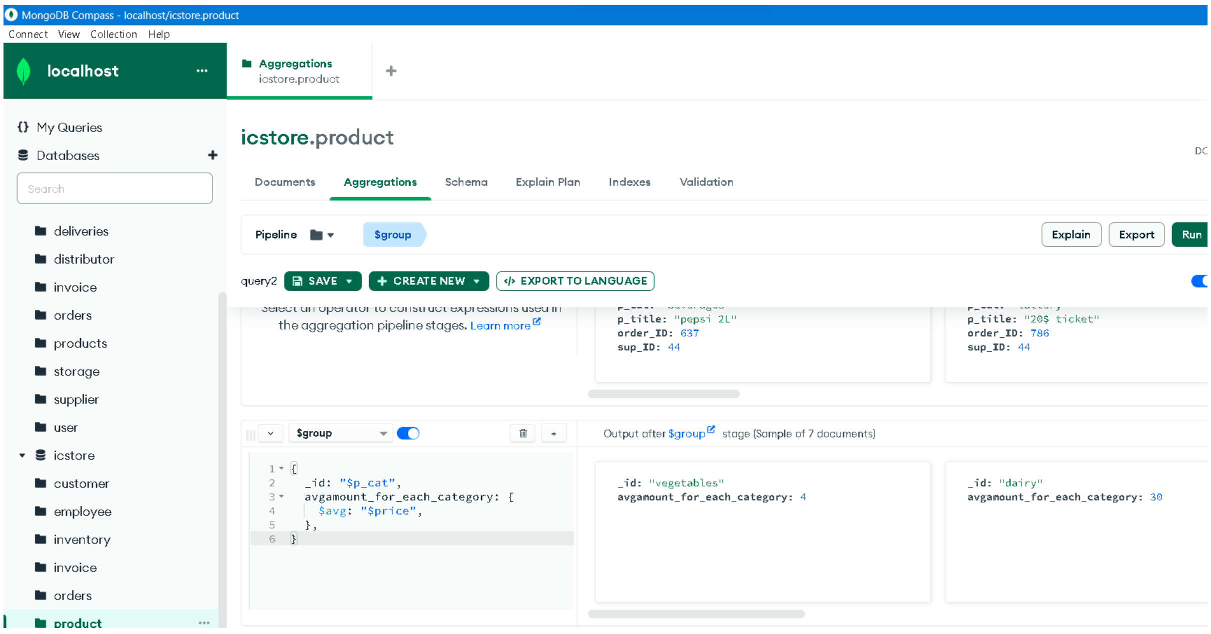
The **$count** stage returns an include of the excess orders in the collection pipeline and relegates the value to a field called *'amount more than 75'*.



```
amount more than 75: 43
```

Ans) There are 43 orders which are above $75.

3) What is the average price of products from each category?



Select an operator to construct expressions used in the aggregation pipeline stages. Learn more

```
p_title: "pepsi 2L"
order_ID: 637
sup_ID: 44
```

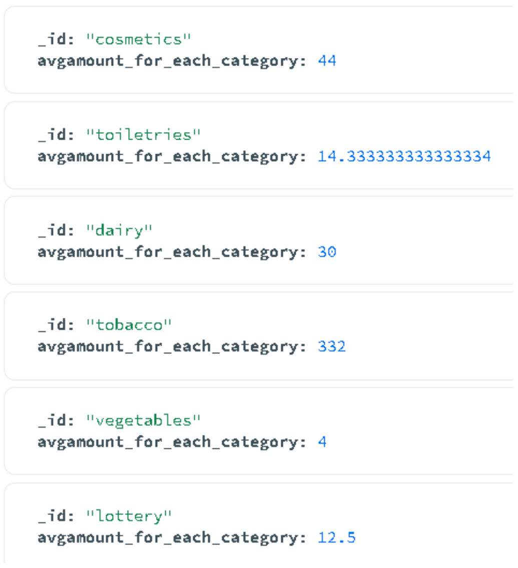
```
p_title: "20$ ticket"
order_ID: 786
sup_ID: 44
```

```
{
  _id: "$p_cat",
  avgamount_for_each_category: {
    $avg: "$price",
  },
}
```

```
_id: "vegetables"
avgamount_for_each_category: 4
```

```
_id: "dairy"
avgamount_for_each_category: 30
```

Grouping the documents by the ‘category’, the following operation uses the **$avg** accumulator to compute the average amount and average price for each grouping.



```
_id: "cosmetics"
avgamount_for_each_category: 44

_id: "toiletries"
avgamount_for_each_category: 14.333333333333334

_id: "dairy"
avgamount_for_each_category: 30

_id: "tobacco"
avgamount_for_each_category: 332

_id: "vegetables"
avgamount_for_each_category: 4

_id: "lottery"
avgamount_for_each_category: 12.5
```