Implementation

For our project I was used two MongoDB software's. Those are

- MongoDB Compass (Contains Databases and Collections).
- MongoDB Shell (Script Development).

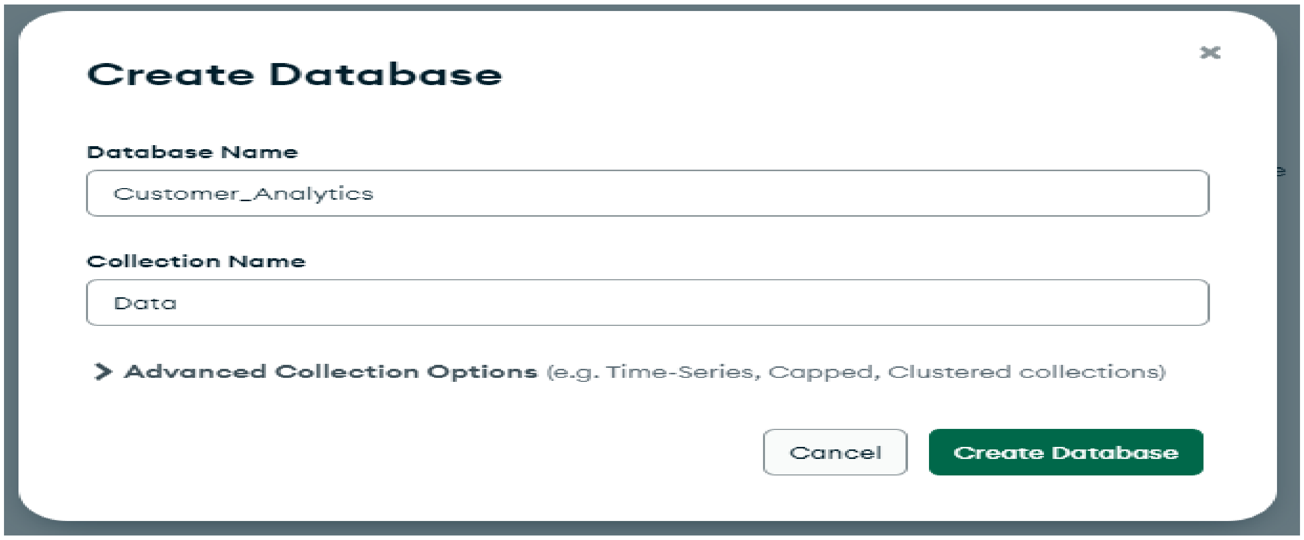
I was used Customer Analytics dataset from Kaggle website. The dataset contains cleaned 10999 observations and 12 attributes.

Here are the clear details of the data.

1. ID: ID is a unique number that is given to the Customers.
2. Gender: Male and Female.
3. Customer rating: The rating is given to the customer to the company on the scale of 1-5 where 1 is lowest and 5 is best.
4. Cost of the product: Cost of the Product in Dollars (US).
5. Mode of transport: The Company transport the products in multiple way such as Ship, Airways and Road.
6. Warehouse block: The Company have big Warehouse and divided in to block such as A, B, C, D, E.
7. Prior purchases: The Number of Prior Purchase.
8. Customer care calls: The number of calls made from enquiry to enquiry of the shipment.
9. Product Importance: The company has categorized into various parameter such as low, medium, high.
10. Discount offered: Discount offered on specific product.
11. Weight in gms: It is the weight in grams.
12. Reached on time: It is the target variable, where 1 Indicates that the product has NOT reached on time and 0 indicates it has reached on time.

By using MongoDB Compass, I was created databases and collections.

Creating a database and collection in Compass:

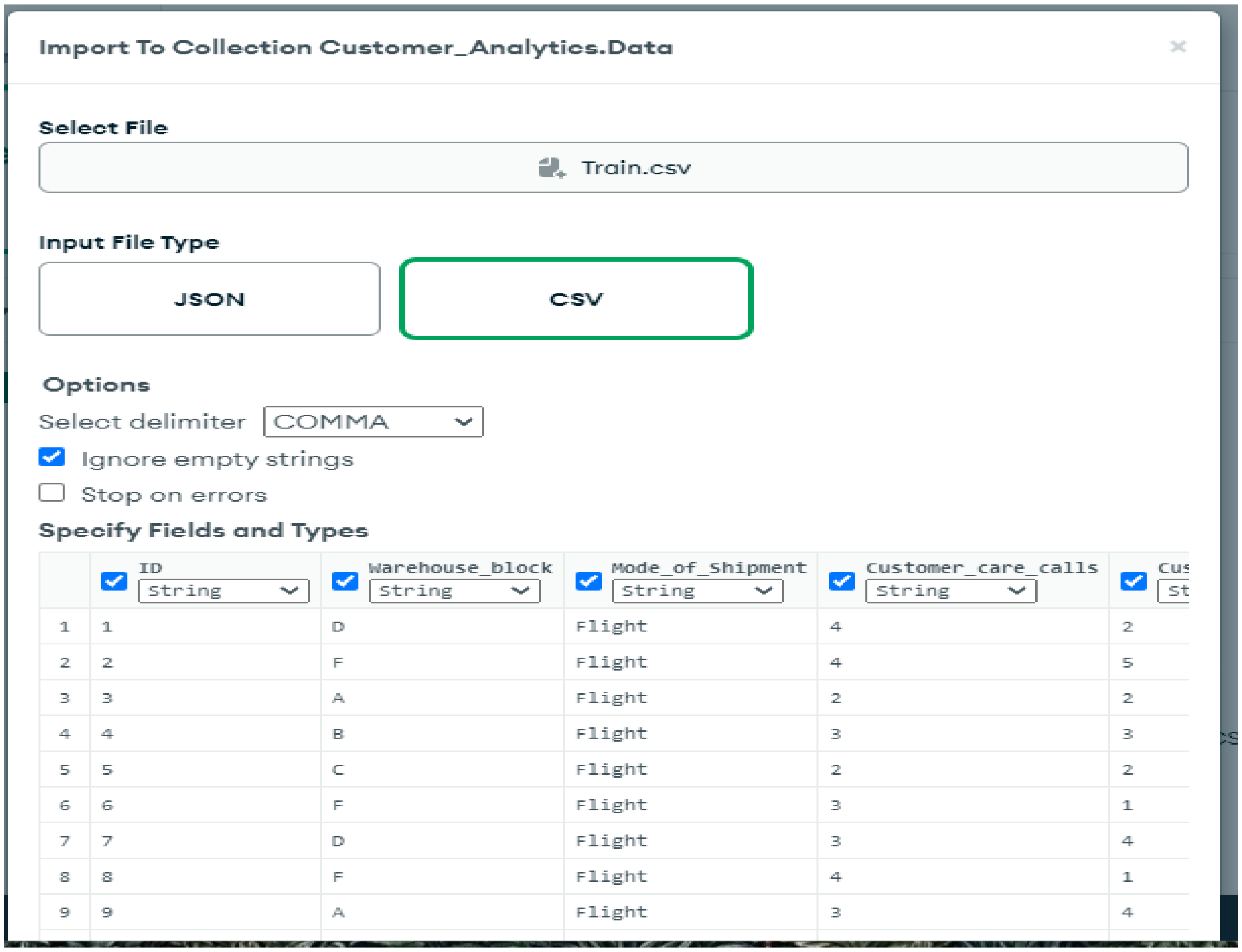


Database Name: Customer_Analytics

Collection Name: Data

Advanced Collection Options (e.g. Time-Series, Capped, Clustered collections)

Importing CSV files to the destination database:



Select File: Train.csv

Input File Type: [✓ CSV]

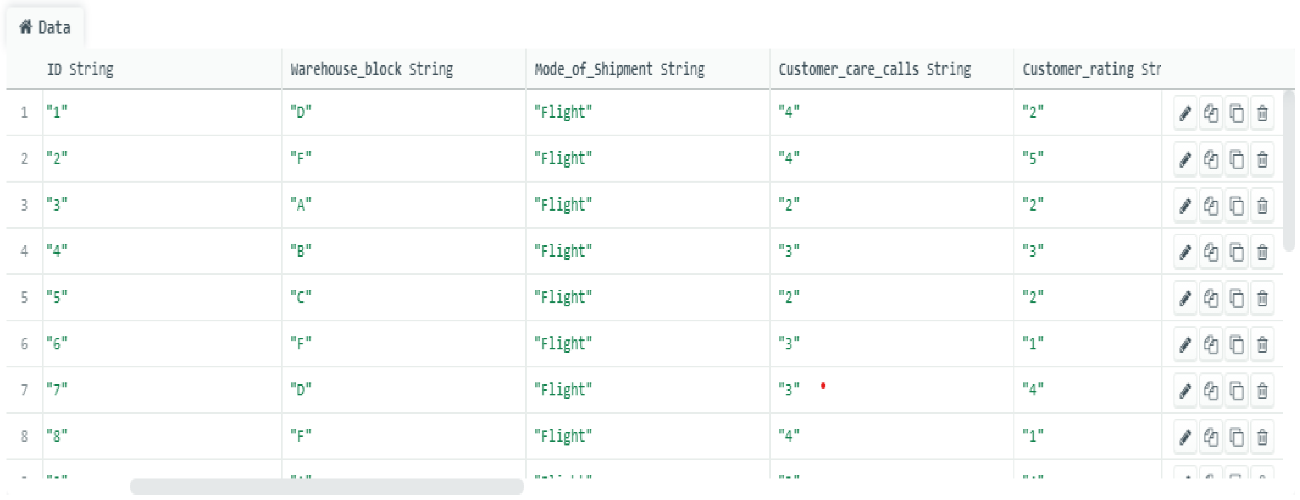
Select delimiter: COMMA

[✓] Ignore empty strings

[ ] Stop on errors

	ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Cus
	String	String	String	String	St
1	1	D	Flight	4	2
2	2	F	Flight	4	5
3	3	A	Flight	2	2
4	4	B	Flight	3	3
5	5	C	Flight	2	2
6	6	F	Flight	3	1
7	7	D	Flight	3	4
8	8	F	Flight	4	1
9	9	A	Flight	3	4

Imported Data:

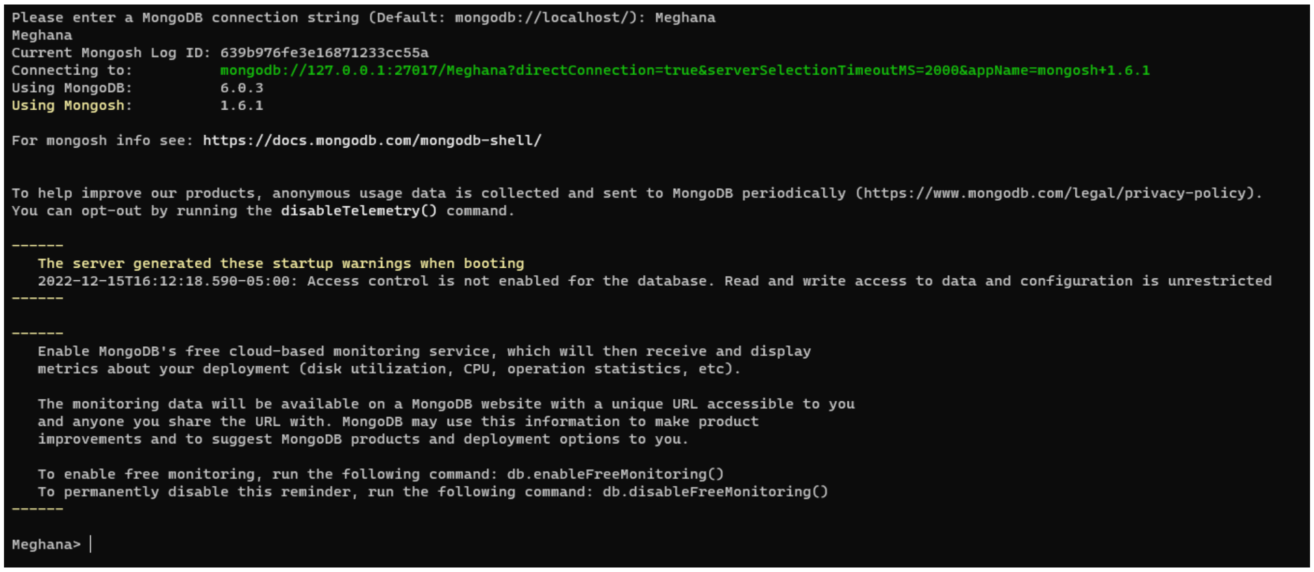


Data

	ID String	Warehouse_block String	Mode_of_Shipment String	Customer_care_calls String	Customer_rating Str
1	"1"	"D"	"Flight"	"4"	"2"
2	"2"	"F"	"Flight"	"4"	"5"
3	"3"	"A"	"Flight"	"2"	"2"
4	"4"	"B"	"Flight"	"3"	"3"
5	"5"	"C"	"Flight"	"2"	"2"
6	"6"	"F"	"Flight"	"3"	"1"
7	"7"	"D"	"Flight"	"3"	"4"
8	"8"	"F"	"Flight"	"4"	"1"

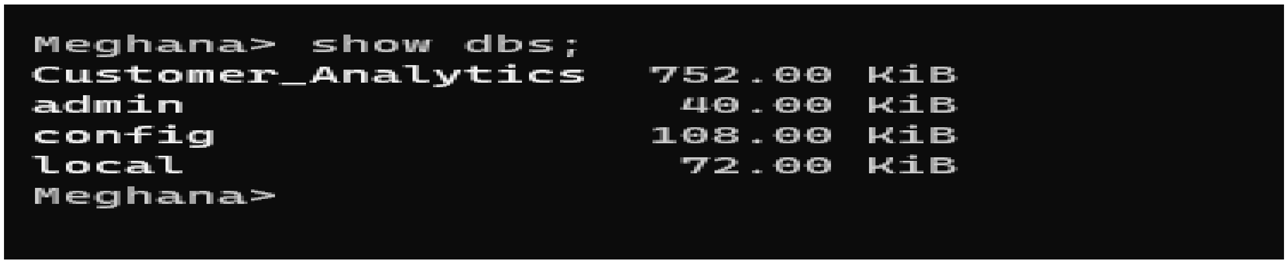
By Using Mongo Shell to develop script for implementation:

Connecting MongoDB Internal Server:



```
Please enter a MongoDB connection string (Default: mongodb://localhost/): Meghana
Meghana
Current Mongosh Log ID: 639b976fe3e16871233cc55a
Connecting to:          mongodb://127.0.0.1:27017/Meghana?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.6.1
Using MongoDB:          6.0.3
Using Mongosh:          1.6.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2022-12-15T16:12:18.590-05:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------

Meghana>
```

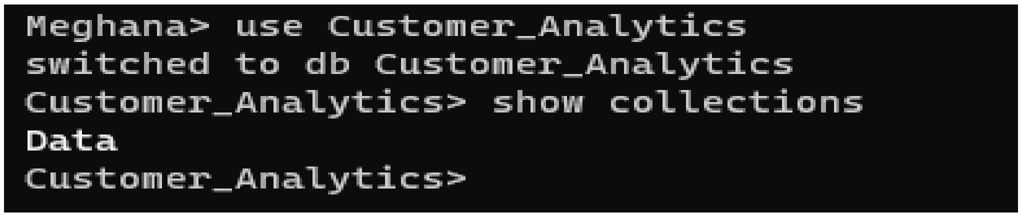
Displaying databases in the MongoDB Database:



```
Meghana> show dbs;
Customer_Analytics   752.00 KiB
admin                 40.00 KiB
config               108.00 KiB
local                 72.00 KiB
Meghana>
```

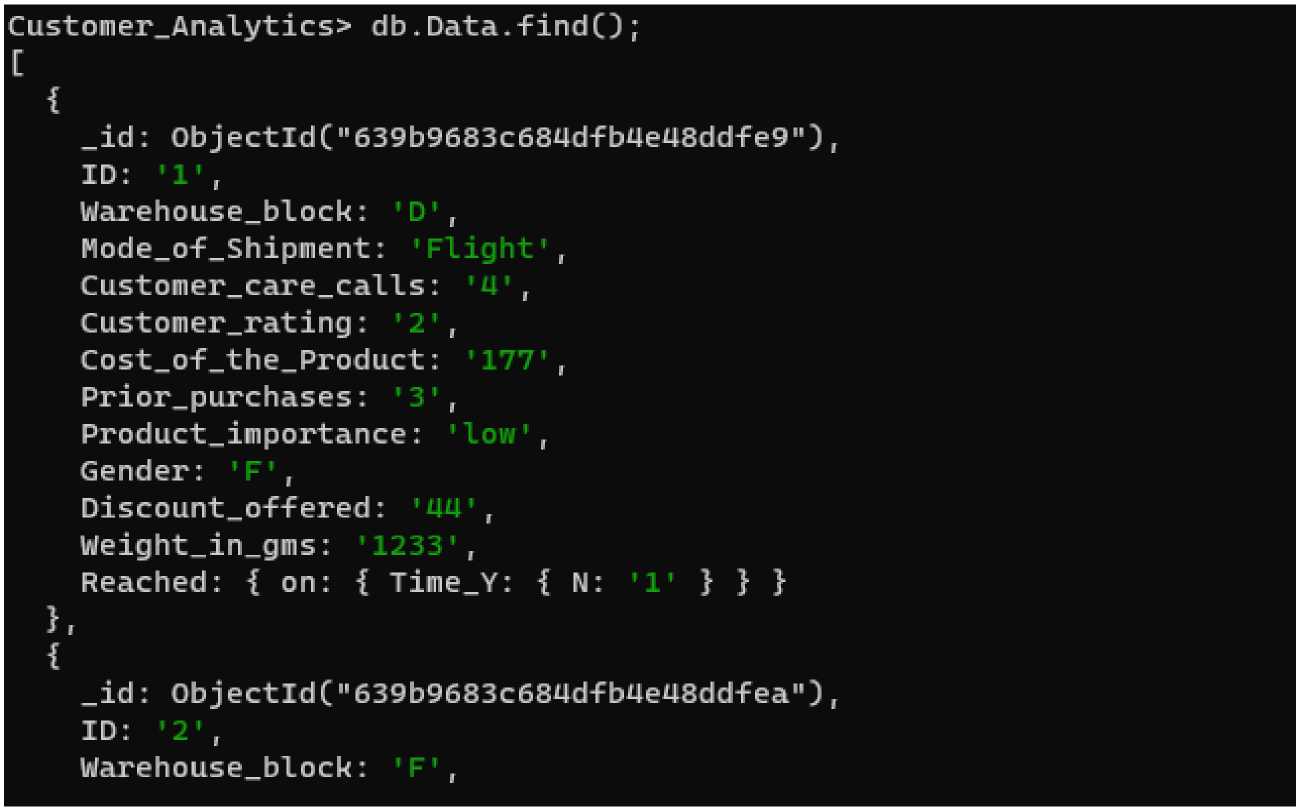
There are 4 Databases in the MongoDB Database.

Using Customer_Analytics database and Displaying collections in that Database:



```
Meghana> use Customer_Analytics
switched to db Customer_Analytics
Customer_Analytics> show collections
Data
Customer_Analytics>
```

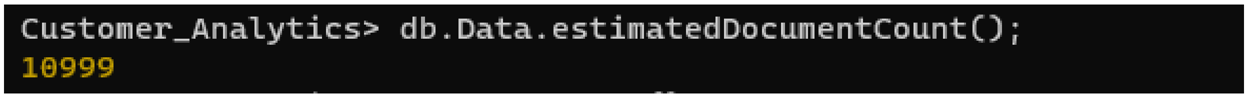
Displaying Data in the collection:



```
Customer_Analytics> db.Data.find();
[
  {
    _id: ObjectId("639b9683c684dfb4e48ddfe9"),
    ID: '1',
    Warehouse_block: 'D',
    Mode_of_Shipment: 'Flight',
    Customer_care_calls: '4',
    Customer_rating: '2',
    Cost_of_the_Product: '177',
    Prior_purchases: '3',
    Product_importance: 'low',
    Gender: 'F',
    Discount_offered: '44',
    Weight_in_gms: '1233',
    Reached: { on: { Time_Y: { N: '1' } } }
  },
  {
    _id: ObjectId("639b9683c684dfb4e48ddfea"),
    ID: '2',
    Warehouse_block: 'F',
```

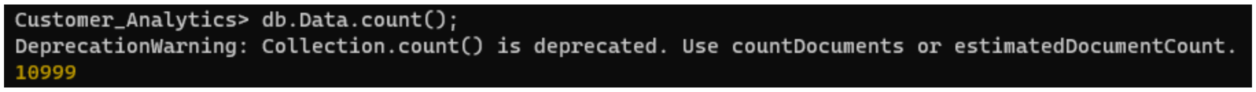
Applying some of the aggregate functions:

To get documents count in the collection:



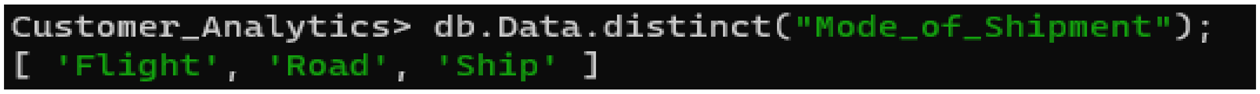
```
Customer_Analytics> db.Data.estimatedDocumentCount();
10999
```

To get number of records in the collection.



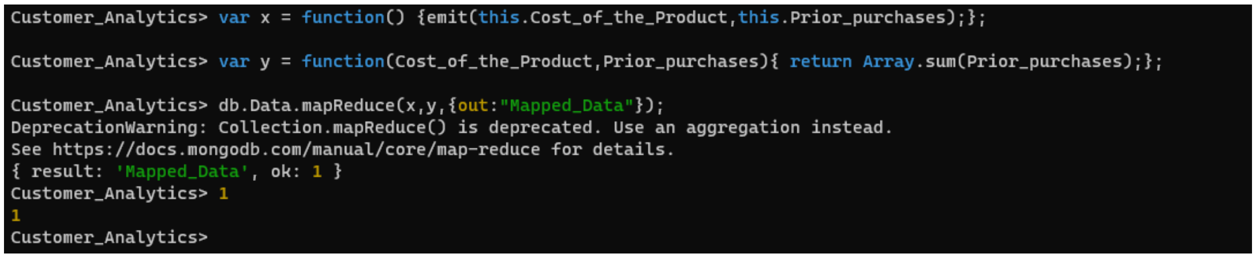
```
Customer_Analytics> db.Data.count();
DeprecationWarning: Collection.count() is deprecated. Use countDocuments or estimatedDocumentCount.
10999
```

To get distinct number of records:



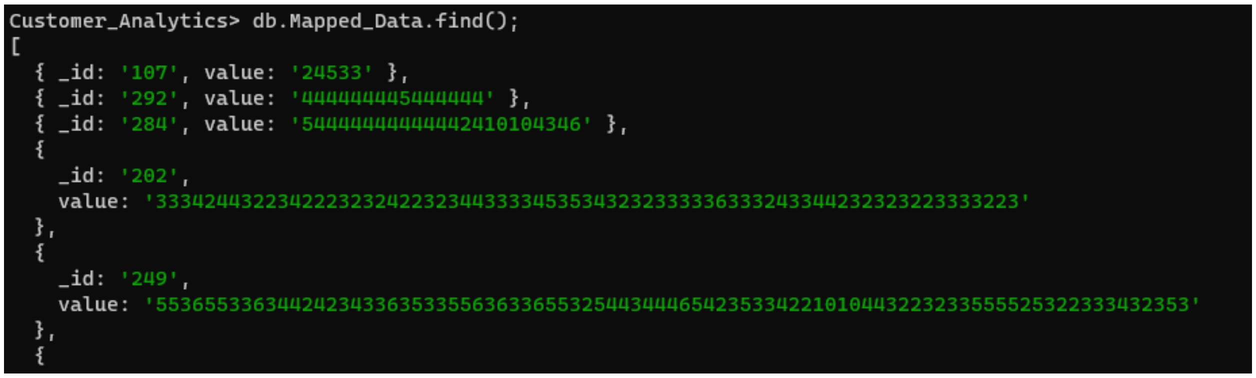
```
Customer_Analytics> db.Data.distinct("Mode_of_Shipment");
[ 'Flight', 'Road', 'Ship' ]
```

Creating MapReduce:



```
Customer_Analytics> var x = function() {emit(this.Cost_of_the_Product,this.Prior_purchases);};

Customer_Analytics> var y = function(Cost_of_the_Product,Prior_purchases){ return Array.sum(Prior_purchases);};

Customer_Analytics> db.Data.mapReduce(x,y,{out:"Mapped_Data"});
DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation instead.
See https://docs.mongodb.com/manual/core/map-reduce for details.
{ result: 'Mapped_Data', ok: 1 }
Customer_Analytics> 1
1
Customer_Analytics>
```

Displaying Mapped_Data:



```
Customer_Analytics> db.Mapped_Data.find();
[
  { _id: '107', value: '24533' },
  { _id: '292', value: '444444445444444' },
  { _id: '284', value: '54444444444442410104346' },
  {
    _id: '202',
    value: '333424432234222323242232344333345353432323333363332433442323232223333223'
  },
  {
    _id: '249',
    value: '5536553363442423433635335563633655325443444654235334221010443223233555525322333432353'
  },
  {
```

Summary

In this Practicum, I observe the below findings.

The practicum main agenda is to prove MongoDB is better than Relational databases. To prove that MongoDB is the best database, I obtained the dataset from the Kaggle website. The dataset was imported and stored in the collection object of the MongoDB Compass. On that dataset, I ran MapReduce and aggregate functions.

The stream processing data from users is mapped or linked using the MapReduce function. To acquire the Exceptional Customer Experience, the data may later be used in dashboards and visualization applications.