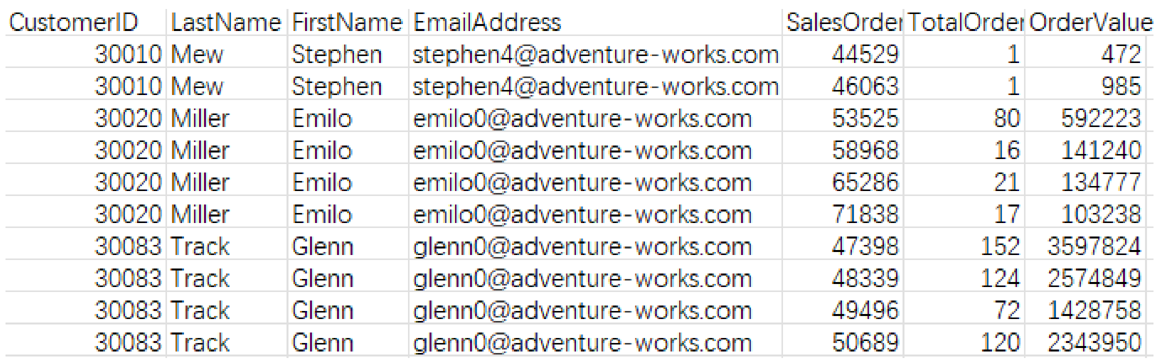
MongoDB-Calculate Average (Using Embedding)

Task goal:

To get all details about a customer using the embedding design technique. With this technique, we'll create one document for each customer. In the customer document, we need to have all the orders of the customer and details about an order. We can use Customer ID as the PK of the customer document by putting it in the _id field.

Example data



CustomerID	LastName	FirstName	EmailAddress	SalesOrde	TotalOrde	OrderValue
30010	Mew	Stephen	stephen4@adventure-works.com	44529	1	472
30010	Mew	Stephen	stephen4@adventure-works.com	46063	1	985
30020	Miller	Emilo	emilo0@adventure-works.com	53525	80	592223
30020	Miller	Emilo	emilo0@adventure-works.com	58968	16	141240
30020	Miller	Emilo	emilo0@adventure-works.com	65286	21	134777
30020	Miller	Emilo	emilo0@adventure-works.com	71838	17	103238
30083	Track	Glenn	glenn0@adventure-works.com	47398	152	3597824
30083	Track	Glenn	glenn0@adventure-works.com	48339	124	2574849
30083	Track	Glenn	glenn0@adventure-works.com	49496	72	1428758
30083	Track	Glenn	glenn0@adventure-works.com	50689	120	2343950

1. MongoDB database design and create documents.

Approach1:

Manually design

- **Understand the requirements, design the schema**
- **Process your raw data, clean it, and transform it into the format that matches your MongoDB schema. For example, using Excel or Google Spreadsheets**
- **Export the processed data to a JSON or CSV file.**
- **Import your JSON or CSV file into MongoDB compass.**
- **After designing manually, go directly to Step 4, skipping Steps 2&3 which are for Approach2 only**

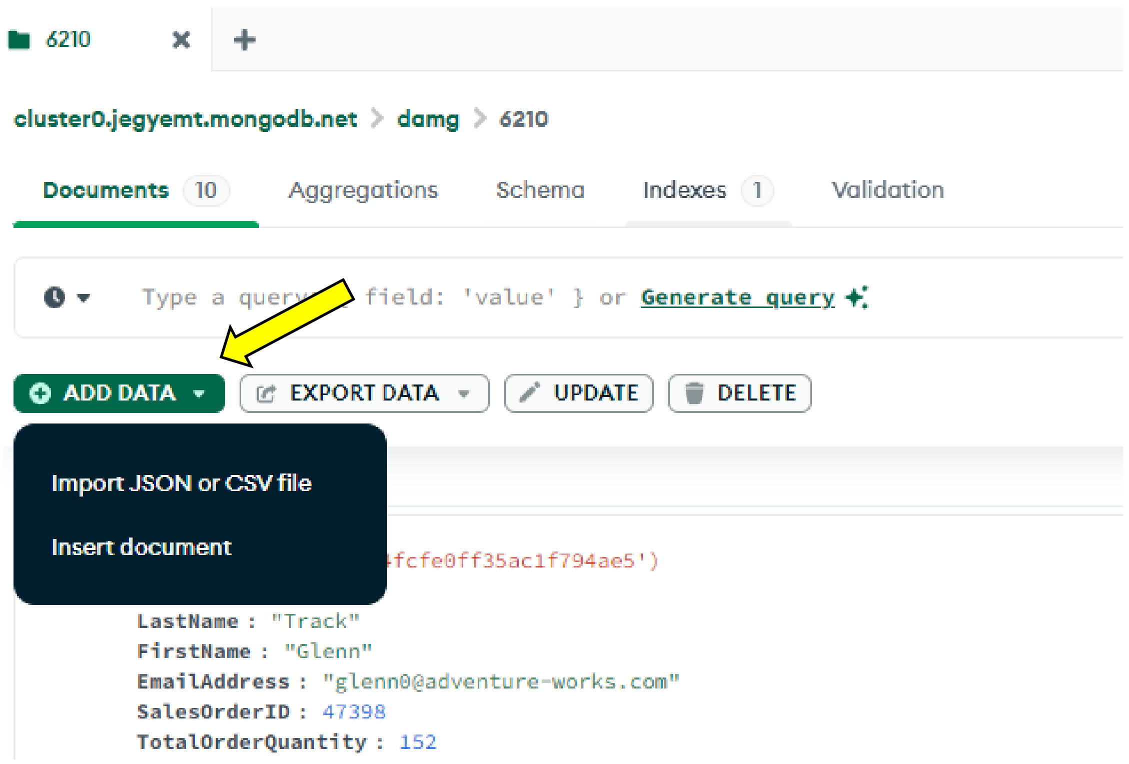
Approach2:

Using Aggregation Pipeline to process your data

- **See Step2&3**

2. Importing Data

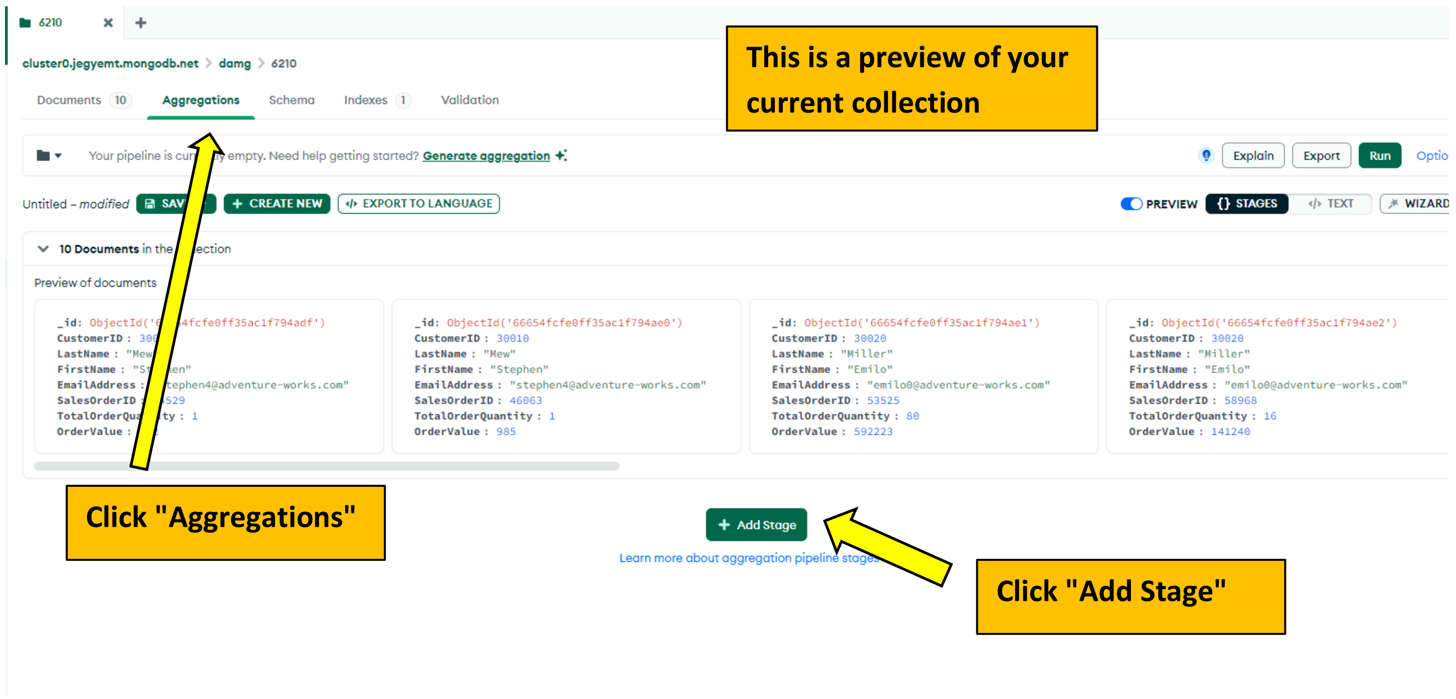
2.1 Open MongoDB Compass and connect to the database and import the raw data (JSON or CSV file), my example database is named "damg" and the collection I inserted the data into is "6210".



```
4fcfe0ff35ac1f794ae5')
    LastName : "Track"
    FirstName : "Glenn"
    EmailAddress : "glenn0@adventure-works.com"
    SalesOrderID : 47398
    TotalOrderQuantity : 152
```

3.Embedding SalesOrderIDs into Customer collection, using Aggregation Pipeline to process raw data

3.1 Select your target database and click on the database to see the collections inside it. You can see a preview of the documents in your collection.

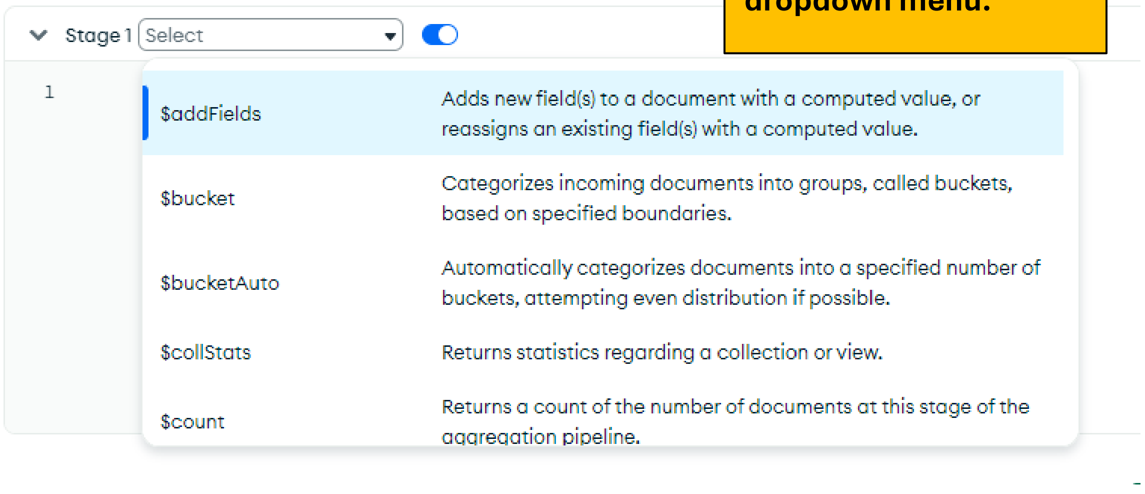

6210
cluster0.jegyemt.mongodb.net > damg > 6210
Documents 10
Aggregations
Schema
Indexes 1
Validation
This is a preview of your current collection
Your pipeline is cur y empty. Need help getting started? Generate aggregation
Explain
Export
Run
Optio
Untitled – modified
SAV
CREATE NEW
EXPORT TO LANGUAGE
PREVIEW
STAGES
TEXT
WIZARD
10 Documents in the ection
Preview of documents
Click "Aggregations"
Add Stage
Learn more about aggregation pipeline stages
Click "Add Stage"

```
_id: ObjectId('6  4fcfe0ff35ac1f794adf')
CustomerID : 300
LastName : "Mew
FirstName : "St   en"
EmailAddress :  tephen4@adventure-works.com"
SalesOrderID :  529
TotalOrderQua   ty : 1
OrderValue :

_id: ObjectId('66654fcfe0ff35ac1f794ae0')
CustomerID : 30010
LastName : "Mew"
FirstName : "Stephen"
EmailAddress : "stephen4@adventure-works.com"
SalesOrderID : 46063
TotalOrderQuantity : 1
OrderValue : 985

_id: ObjectId('66654fcfe0ff35ac1f794ae1')
CustomerID : 30020
LastName : "Miller"
FirstName : "Emilo"
EmailAddress : "emilo0@adventure-works.com"
SalesOrderID : 53525
TotalOrderQuantity : 80
OrderValue : 592223

_id: ObjectId('66654fcfe0ff35ac1f794ae2')
CustomerID : 30020
LastName : "Miller"
FirstName : "Emilo"
EmailAddress : "emilo0@adventure-works.com"
SalesOrderID : 58968
TotalOrderQuantity : 16
OrderValue : 141240
```

3.2 Navigate to the "Aggregations" tab within your selected collection, this will allow you to start creating an aggregation pipeline. Click on the "Create New" button to start building a new aggregation pipeline.

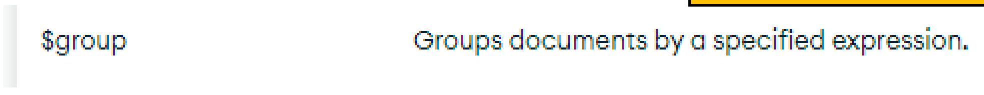
3.3 Click on the "Stage 1" dropdown menu and select $group from the list of aggregation stages.

click on the "Stage 1" dropdown menu.

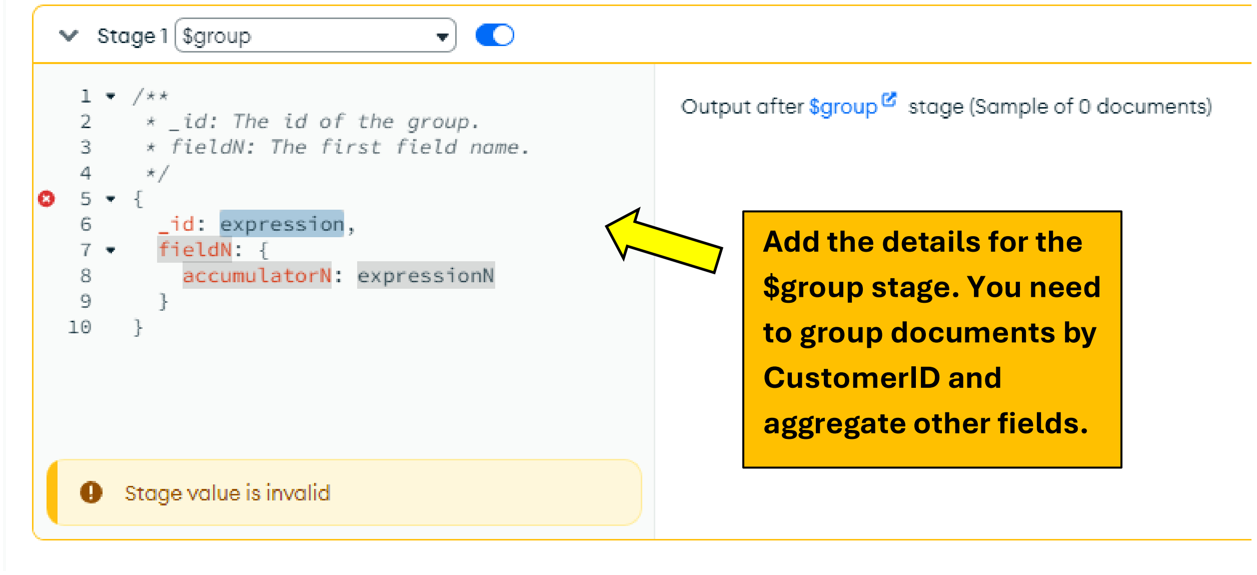


$addFields	Adds new field(s) to a document with a computed value, or reassigns an existing field(s) with a computed value.
$bucket	Categorizes incoming documents into groups, called buckets, based on specified boundaries.
$bucketAuto	Automatically categorizes documents into a specified number of buckets, attempting even distribution if possible.
$collStats	Returns statistics regarding a collection or view.
$count	Returns a count of the number of documents at this stage of the aggregation pipeline.

select $group from the list of aggregation



$group	Groups documents by a specified expression.



```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: expression,
  fieldN: {
    accumulatorN: expressionN
  }
}
```

Stage value is invalid

Output after $group stage (Sample of 0 documents)

Add the details for the $group stage. You need to group documents by CustomerID and aggregate other fields.

3.4 Define the Grouping Key:

_id: Set this to "$CustomerID" to group by CustomerID.

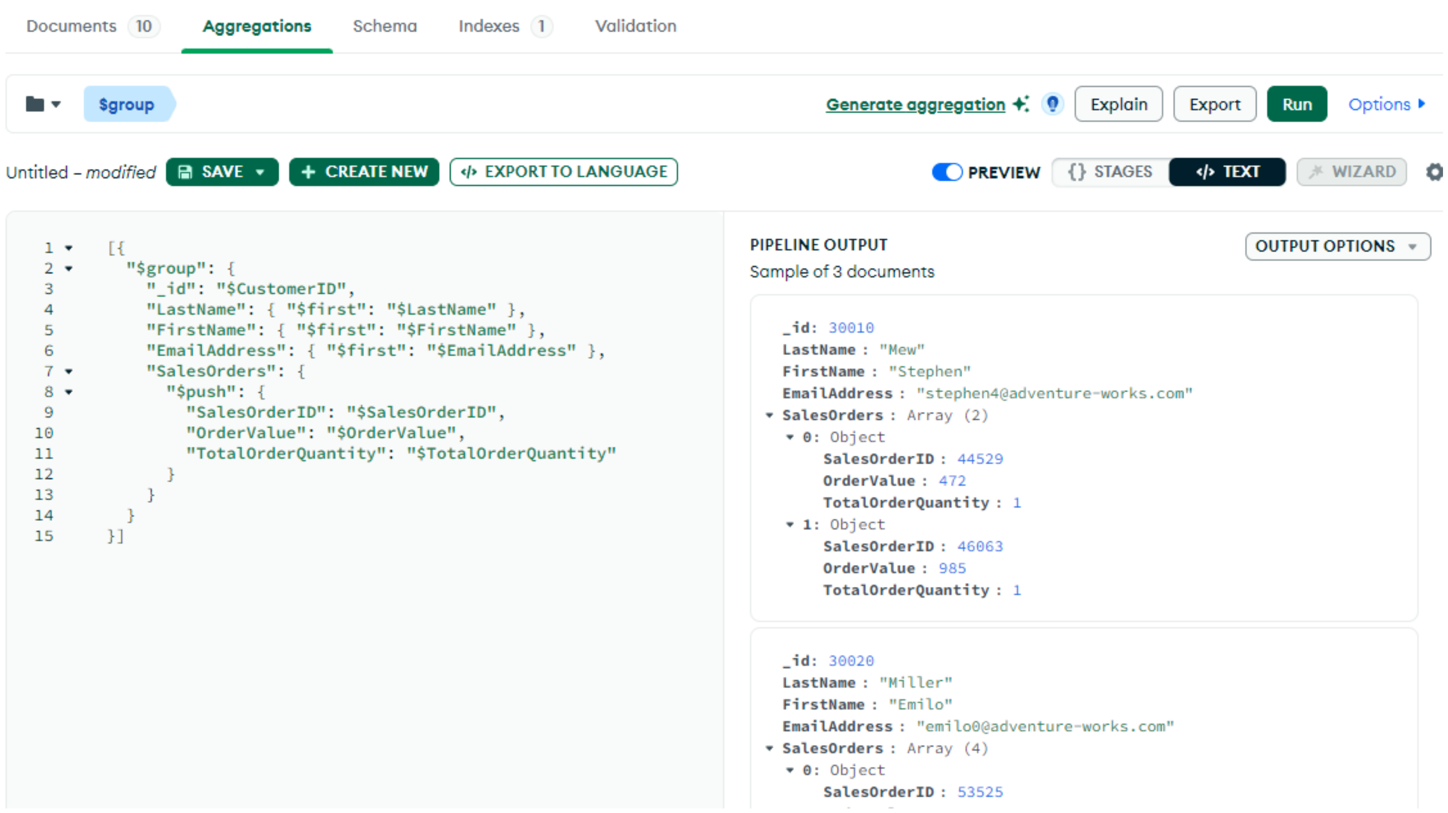
Aggregate Fields:

LastName: Use $first to take the first occurrence of LastName.

FirstName: Use $first to take the first occurrence of FirstName.

EmailAddress: Use $first to take the first occurrence of EmailAddress.

SalesOrderID: Use $push to aggregate SalesOrderID, OrderValue, and TotalOrderQuantity into an array.

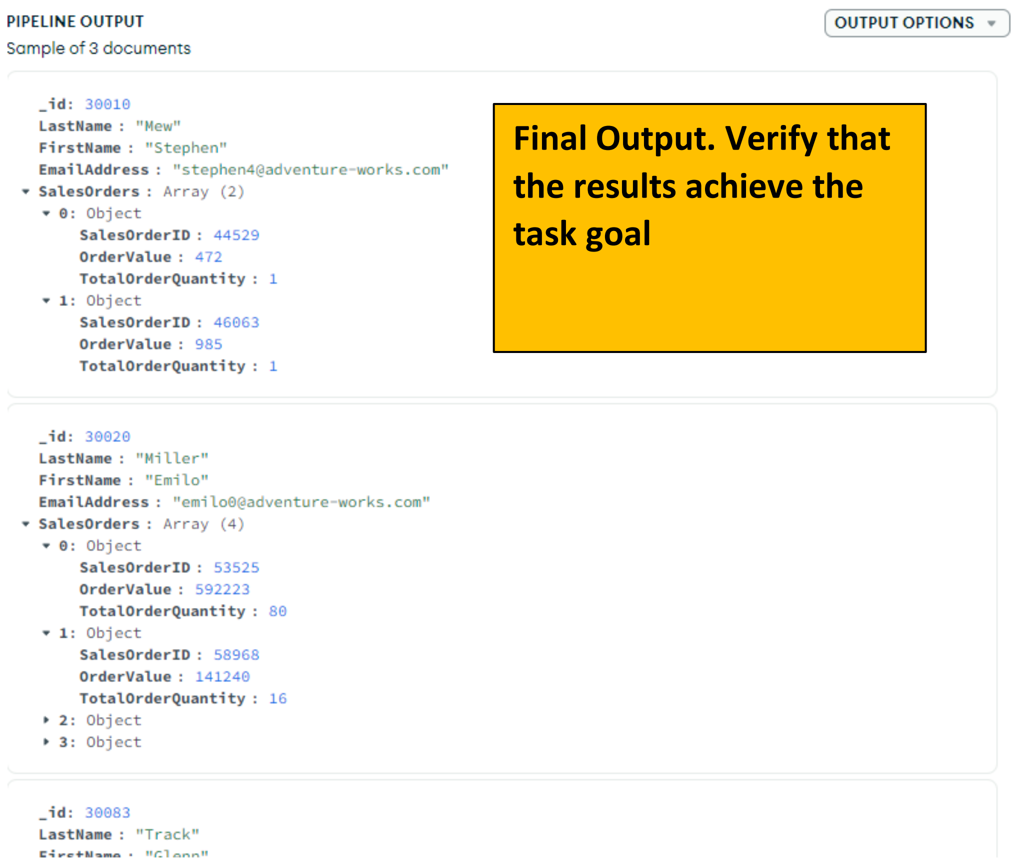


```
[{
  "$group": {
    "_id": "$CustomerID",
    "LastName": { "$first": "$LastName" },
    "FirstName": { "$first": "$FirstName" },
    "EmailAddress": { "$first": "$EmailAddress" },
    "SalesOrders": {
      "$push": {
        "SalesOrderID": "$SalesOrderID",
        "OrderValue": "$OrderValue",
        "TotalOrderQuantity": "$TotalOrderQuantity"
      }
    }
  }
}]
```

Sample of 3 documents

```
_id: 30010
LastName : "Mew"
FirstName : "Stephen"
EmailAddress : "stephen4@adventure-works.com"
SalesOrders : Array (2)
  0: Object
    SalesOrderID : 44529
    OrderValue : 472
    TotalOrderQuantity : 1
  1: Object
    SalesOrderID : 46063
    OrderValue : 985
    TotalOrderQuantity : 1

_id: 30020
LastName : "Miller"
FirstName : "Emilo"
EmailAddress : "emilo0@adventure-works.com"
SalesOrders : Array (4)
  0: Object
    SalesOrderID : 53525
```

3.5 Final Output, examine the Aggregation Results, check the final output to ensure that it includes all the required fields.


PIPELINE OUTPUT

Sample of 3 documents

OUTPUT OPTIONS

```
  _id: 30010
  LastName : "Mew"
  FirstName : "Stephen"
  EmailAddress : "stephen4@adventure-works.com"
▾ SalesOrders : Array (2)
  ▾ 0: Object
      SalesOrderID : 44529
      OrderValue : 472
      TotalOrderQuantity : 1
  ▾ 1: Object
      SalesOrderID : 46063
      OrderValue : 985
      TotalOrderQuantity : 1
```

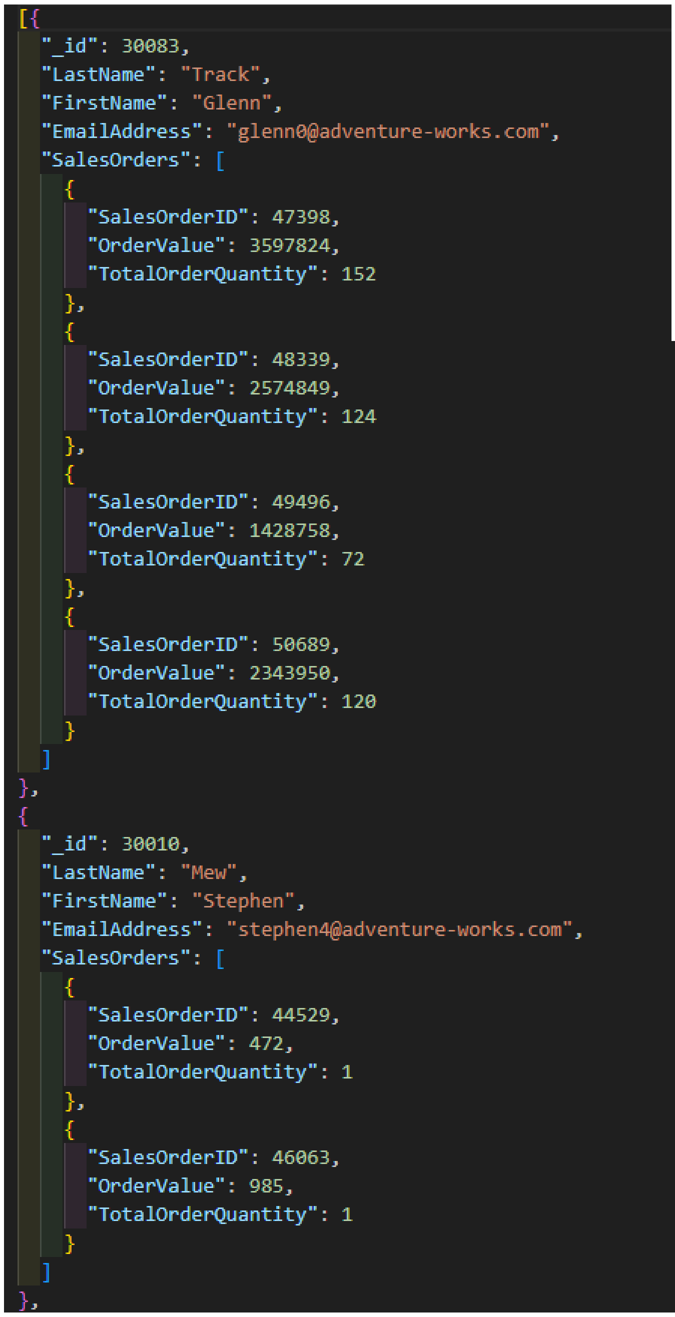
Final Output. Verify that the results achieve the task goal

```
  _id: 30020
  LastName : "Miller"
  FirstName : "Emilo"
  EmailAddress : "emilo0@adventure-works.com"
▾ SalesOrders : Array (4)
  ▾ 0: Object
      SalesOrderID : 53525
      OrderValue : 592223
      TotalOrderQuantity : 80
  ▾ 1: Object
      SalesOrderID : 58968
      OrderValue : 141240
      TotalOrderQuantity : 16
  ▸ 2: Object
  ▸ 3: Object
```

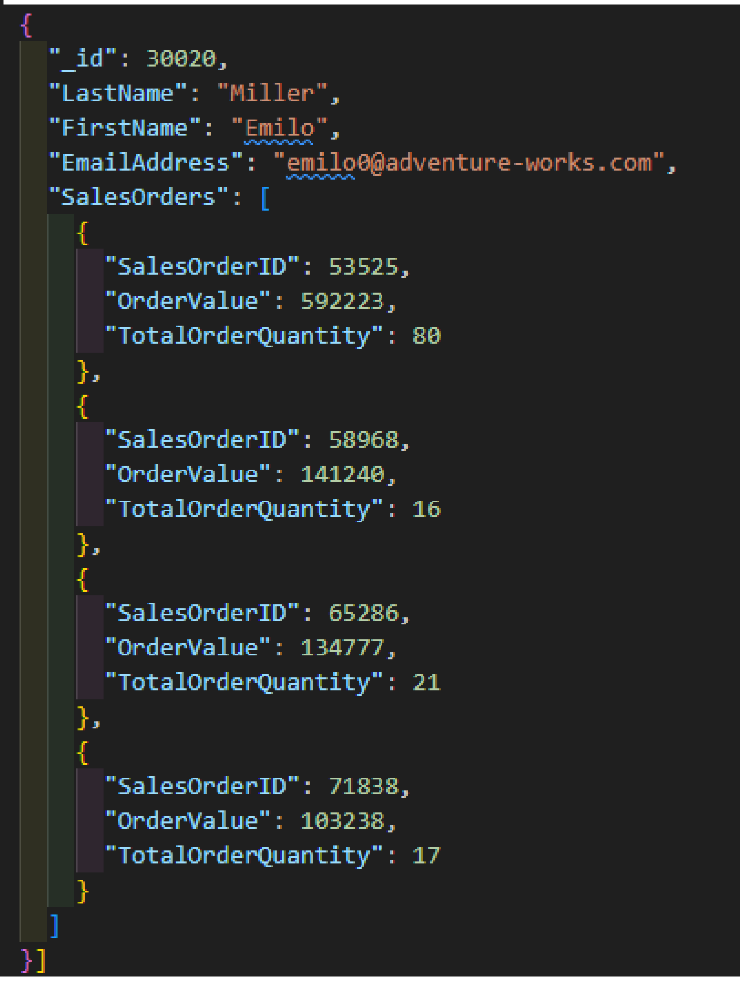
```
  _id: 30083
  LastName : "Track"
  FirstName : "Glenn"
```

<u>4. Before Calculation, verify your processed data again (Common to both Approach1 and Approach 2)</u>

Processed data collection in JSON format



```
[{
  "_id": 30083,
  "LastName": "Track",
  "FirstName": "Glenn",
  "EmailAddress": "glenn0@adventure-works.com",
  "SalesOrders": [
    {
      "SalesOrderID": 47398,
      "OrderValue": 3597824,
      "TotalOrderQuantity": 152
    },
    {
      "SalesOrderID": 48339,
      "OrderValue": 2574849,
      "TotalOrderQuantity": 124
    },
    {
      "SalesOrderID": 49496,
      "OrderValue": 1428758,
      "TotalOrderQuantity": 72
    },
    {
      "SalesOrderID": 50689,
      "OrderValue": 2343950,
      "TotalOrderQuantity": 120
    }
  ]
},
{
  "_id": 30010,
  "LastName": "Mew",
  "FirstName": "Stephen",
  "EmailAddress": "stephen4@adventure-works.com",
  "SalesOrders": [
    {
      "SalesOrderID": 44529,
      "OrderValue": 472,
      "TotalOrderQuantity": 1
    },
    {
      "SalesOrderID": 46063,
      "OrderValue": 985,
      "TotalOrderQuantity": 1
    }
  ]
},
```



```
{
  "_id": 30020,
  "LastName": "Miller",
  "FirstName": "Emilo",
  "EmailAddress": "emilo0@adventure-works.com",
  "SalesOrders": [
    {
      "SalesOrderID": 53525,
      "OrderValue": 592223,
      "TotalOrderQuantity": 80
    },
    {
      "SalesOrderID": 58968,
      "OrderValue": 141240,
      "TotalOrderQuantity": 16
    },
    {
      "SalesOrderID": 65286,
      "OrderValue": 134777,
      "TotalOrderQuantity": 21
    },
    {
      "SalesOrderID": 71838,
      "OrderValue": 103238,
      "TotalOrderQuantity": 17
    }
  ]
}]
```

5.Calculate average using Aggregation Pipeline (Common to both Approach1 and Approach 2)

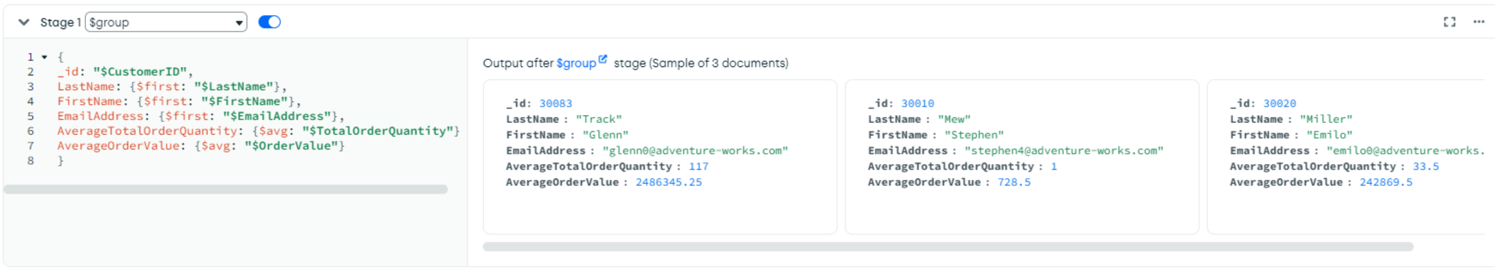
5.1 Define the Grouping Key:

_id: Set this to "$CustomerID" to group by CustomerID.

Aggregate Fields:

AverageTotalOrderQuantity: Use $avg to calculate the average of TotalOrderQuantity.

AverageOrderValue: Use $avg to calculate the average of OrderValue.


Stage 1 $group

```
{
_id: "$CustomerID",
LastName: {$first: "$LastName"},
FirstName: {$first: "$FirstName"},
EmailAddress: {$first: "$EmailAddress"},
AverageTotalOrderQuantity: {$avg: "$TotalOrderQuantity"}
AverageOrderValue: {$avg: "$OrderValue"}
}
```

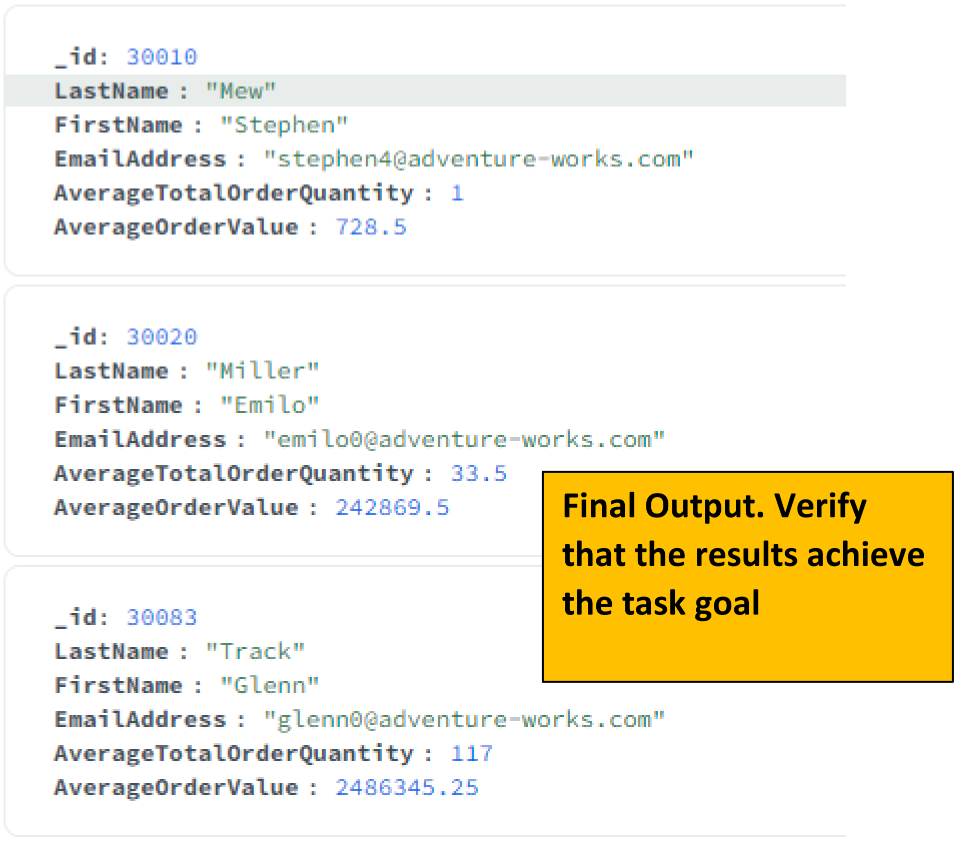
Output after $group stage (Sample of 3 documents)

```
_id: 30083
LastName : "Track"
FirstName : "Glenn"
EmailAddress : "glenn0@adventure-works.com"
AverageTotalOrderQuantity : 117
AverageOrderValue : 2486345.25
```

```
_id: 30010
LastName : "Mew"
FirstName : "Stephen"
EmailAddress : "stephen4@adventure-works.com"
AverageTotalOrderQuantity : 1
AverageOrderValue : 728.5
```

```
_id: 30020
LastName : "Miller"
FirstName : "Emilo"
EmailAddress : "emilo0@adventure-works.
AverageTotalOrderQuantity : 33.5
AverageOrderValue : 242869.5
```

5.2 Final Output, examine the Aggregation Results,verify that the aggregation has correctly grouped the documents by CustomerID and calculated the average values as needed.



```
_id: 30010
LastName : "Mew"
FirstName : "Stephen"
EmailAddress : "stephen4@adventure-works.com"
AverageTotalOrderQuantity : 1
AverageOrderValue : 728.5
```

```
_id: 30020
LastName : "Miller"
FirstName : "Emilo"
EmailAddress : "emilo0@adventure-works.com"
AverageTotalOrderQuantity : 33.5
AverageOrderValue : 242869.5
```

```
_id: 30083
LastName : "Track"
FirstName : "Glenn"
EmailAddress : "glenn0@adventure-works.com"
AverageTotalOrderQuantity : 117
AverageOrderValue : 2486345.25
```

Final Output. Verify that the results achieve the task goal