Name:Siddarth
Date:31-07-2024

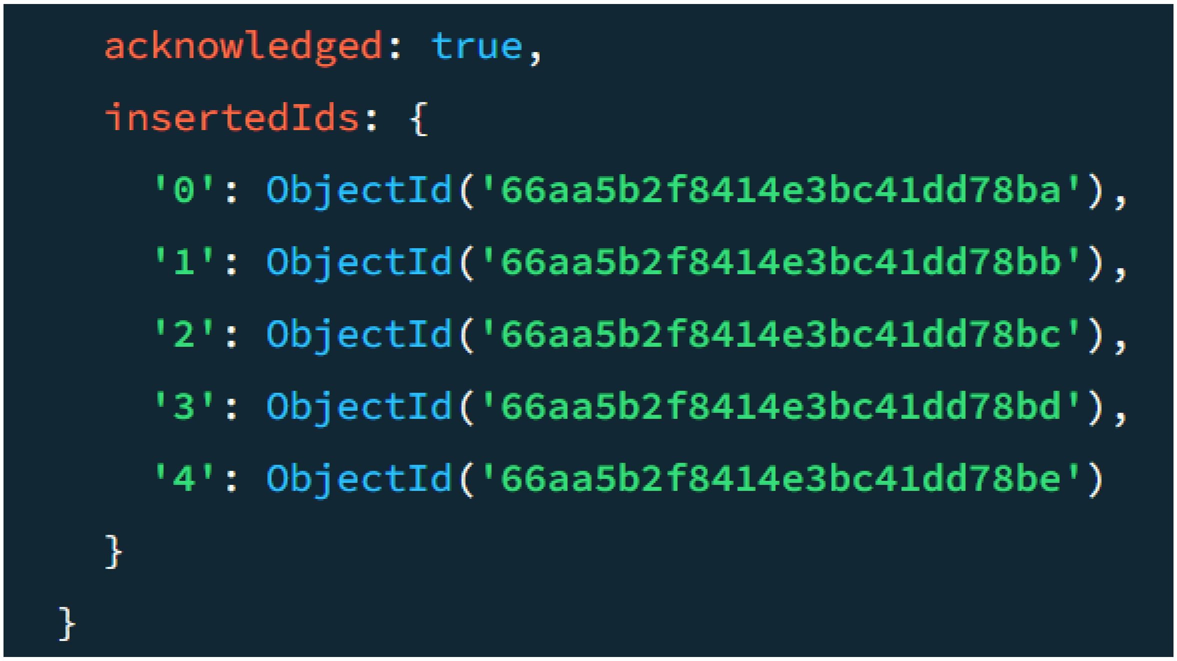
Assignment 2: Aggregation Framework Objective: Understand and use the aggregation framework to perform complex data analysis.

Tasks: Insert documents into a sales collection with fields such as item, quantity, price, and date. Write aggregation pipelines to: Calculate the total sales amount for each item. Find the average quantity sold per item. Group sales by month and calculate the total sales for each month and sort from the largest value Display which year has the maximum sales

Solution:
1.Insert Documents into a Sales Collection
Query:

```
db.sales.insertMany([
   { item: "Rice Bag", quantity: 50, price: 500, date:
new Date("2023-01-15") },
   { item: "Coconut Oil", quantity: 20, price: 150,
date: new Date("2023-02-20") },
   { item: "Dosa Batter", quantity: 100, price: 50,
date: new Date("2023-03-25") },
   { item: "Idli Rice", quantity: 75, price: 60, date:
new Date("2023-04-18") },
   { item: "Urad Dal", quantity: 40, price: 80, date:
new Date("2023-05-22") }
]);
```

Output:



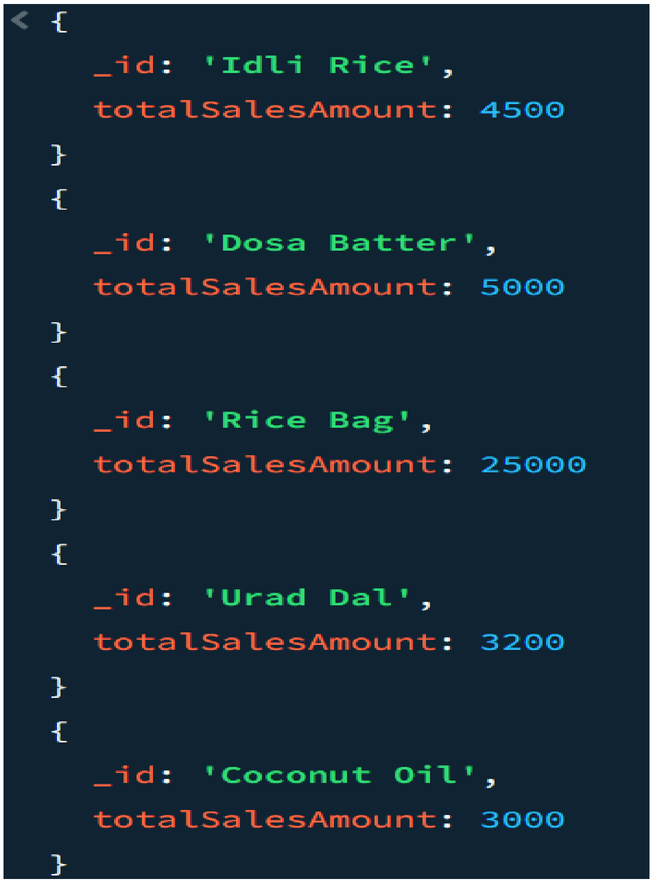
```
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('66aa5b2f8414e3bc41dd78ba'),
      '1': ObjectId('66aa5b2f8414e3bc41dd78bb'),
      '2': ObjectId('66aa5b2f8414e3bc41dd78bc'),
      '3': ObjectId('66aa5b2f8414e3bc41dd78bd'),
      '4': ObjectId('66aa5b2f8414e3bc41dd78be')
    }
  }
```

2.Calculate the Total Sales Amount for Each Item

Query:

```
db.sales.aggregate([
   {
      $group: {
         _id: "$item",
         totalSalesAmount: { $sum: { $multiply:
["$quantity", "$price"] } }
      }
   }
]);
```

Output:



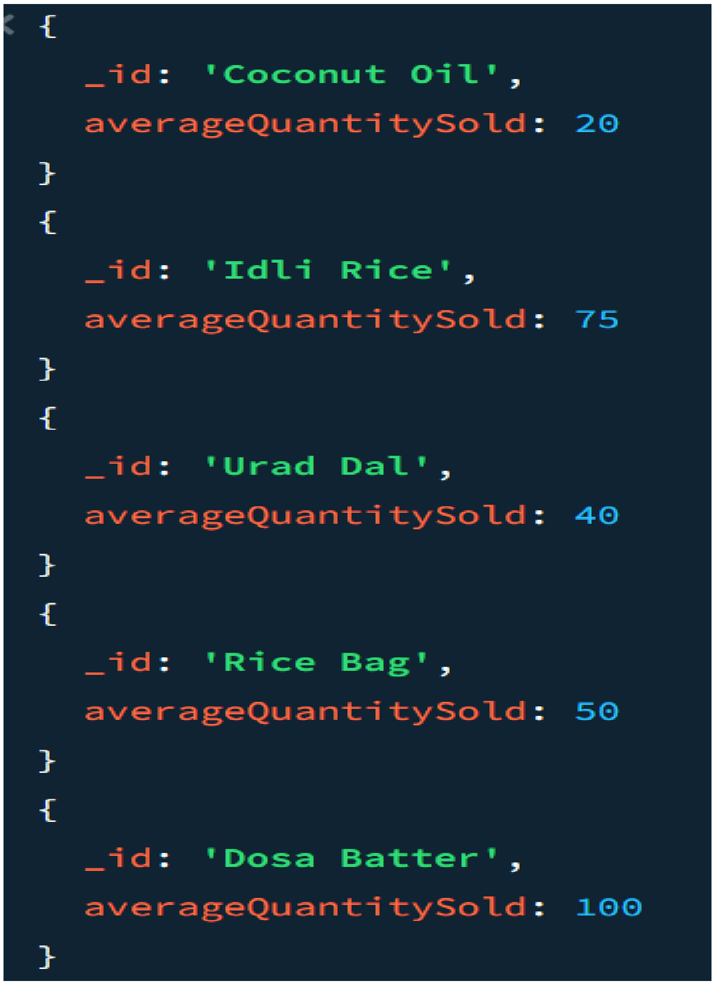
```
< {
    _id: 'Idli Rice',
    totalSalesAmount: 4500
  }
  {
    _id: 'Dosa Batter',
    totalSalesAmount: 5000
  }
  {
    _id: 'Rice Bag',
    totalSalesAmount: 25000
  }
  {
    _id: 'Urad Dal',
    totalSalesAmount: 3200
  }
  {
    _id: 'Coconut Oil',
    totalSalesAmount: 3000
  }
```

3.Find the Average Quantity Sold Per Item

Query:

```
db.sales.aggregate([
   {
      $group: {
         _id: "$item",
         averageQuantitySold: { $avg: "$quantity" }
      }
   }
]);
```

Output:



```
{
  _id: 'Coconut Oil',
  averageQuantitySold: 20
}
{
  _id: 'Idli Rice',
  averageQuantitySold: 75
}
{
  _id: 'Urad Dal',
  averageQuantitySold: 40
}
{
  _id: 'Rice Bag',
  averageQuantitySold: 50
}
{
  _id: 'Dosa Batter',
  averageQuantitySold: 100
}
```

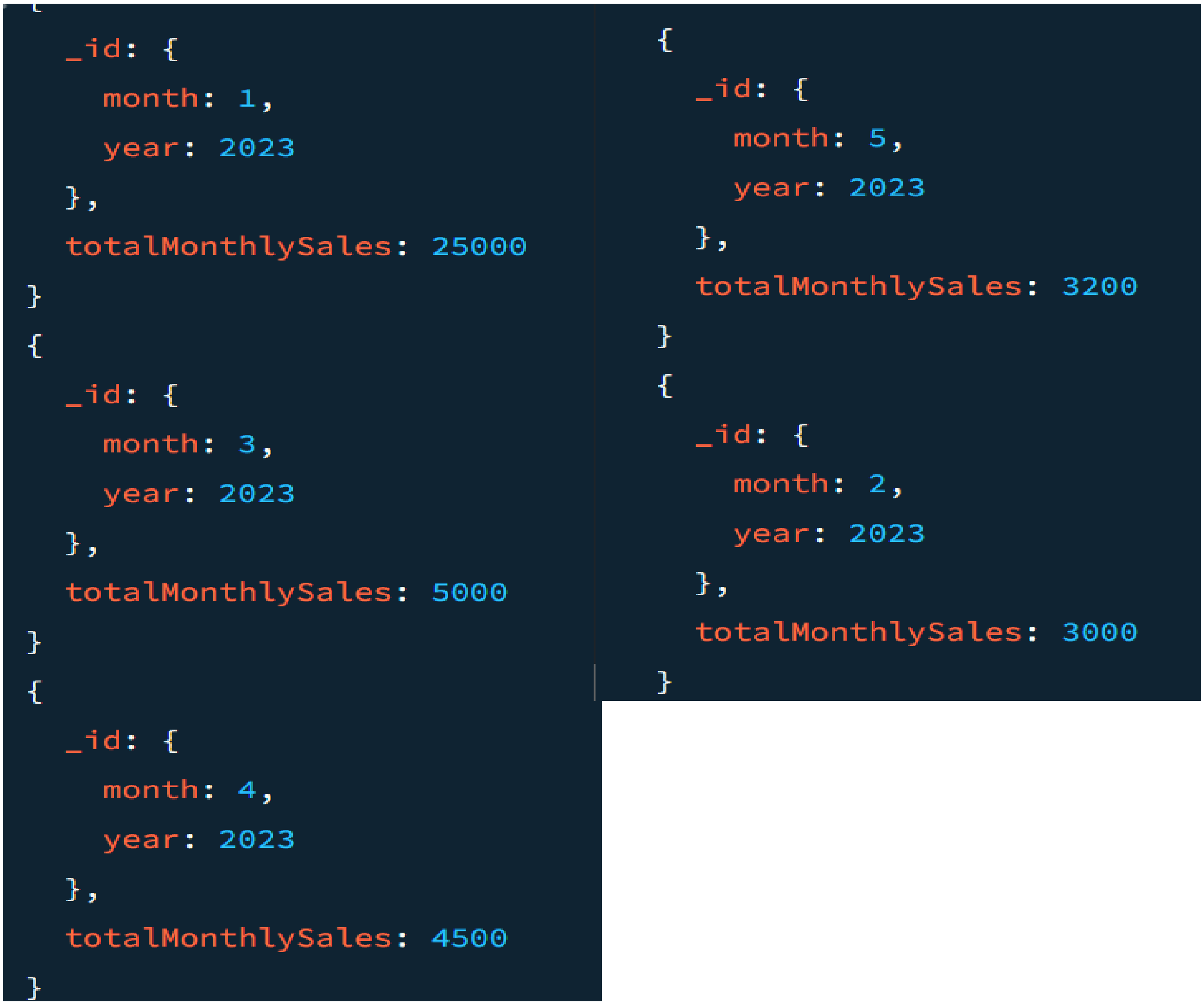
4.Group Sales by Month and Calculate the Total Sales for Each Month, Then Sort from Largest Value

Query:

```
db.sales.aggregate([
   {
      $group: {
         _id: { month: { $month: "$date" }, year: {
$year: "$date" } },
         totalMonthlySales: { $sum: { $multiply:
["$quantity", "$price"] } }
      }
```

```
    },
    {
        $sort: { totalMonthlySales: -1 }
    }
]);
```

Output:



```
{
  _id: {
    month: 1,
    year: 2023
  },
  totalMonthlySales: 25000
}
{
  _id: {
    month: 3,
    year: 2023
  },
  totalMonthlySales: 5000
}
{
  _id: {
    month: 4,
    year: 2023
  },
  totalMonthlySales: 4500
}
```

```
{
  _id: {
    month: 5,
    year: 2023
  },
  totalMonthlySales: 3200
}
{
  _id: {
    month: 2,
    year: 2023
  },
  totalMonthlySales: 3000
}
```

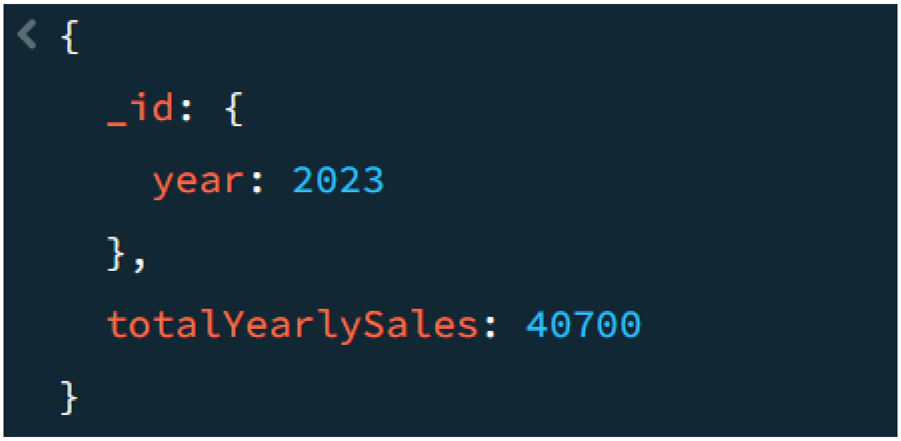
5.Display Which Year Has the Maximum Sales

Query:

```
db.sales.aggregate([
   {
      $group: {
         _id: { year: { $year: "$date" } },
         totalYearlySales: { $sum: { $multiply:
["$quantity", "$price"] } }
      }
   },
   {
      $sort: { totalYearlySales: -1 }
   },
   {
      $limit: 1
   }

    ]);
```

Output:



```
< {
    _id: {
      year: 2023
    },
    totalYearlySales: 40700
  }
```