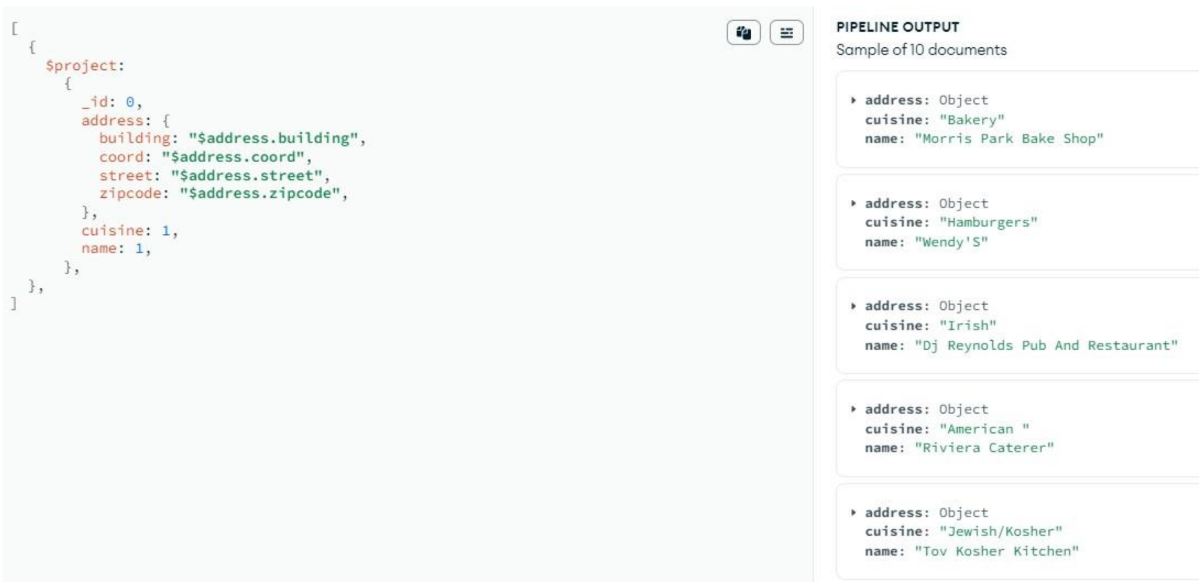
1. Write a query statement to display only *address*, *cuisine* and *name*.

```
[
  {
   $project:
     {
       _id: 0,
       address: {
         building: "$address.building",
         coord: "$address.coord",
         street: "$address.street",
         zipcode: "$address.zipcode",
       },
       cuisine: 1,
       name: 1,
     },
  },
]
```



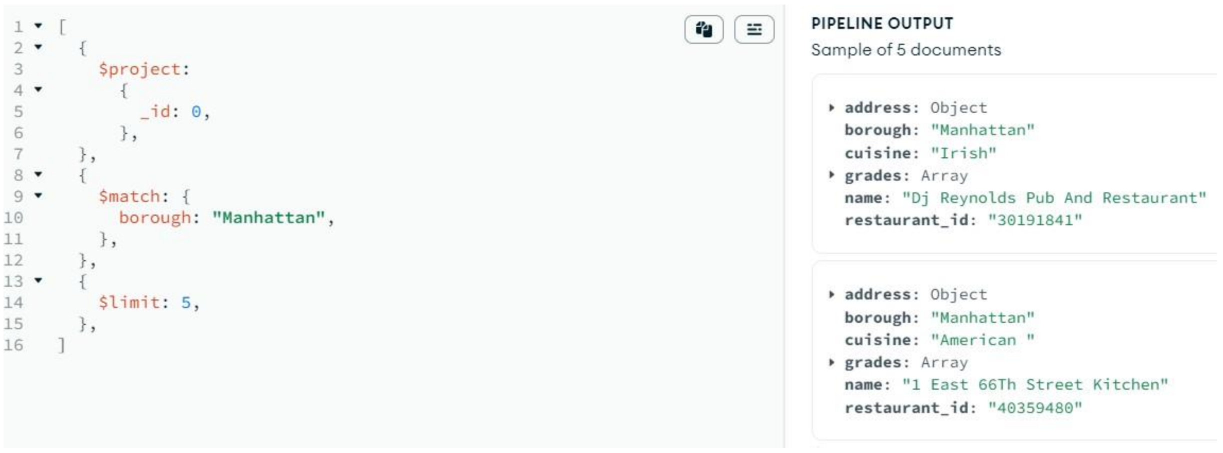
```
[
  {
    $project:
      {
        _id: 0,
        address: {
          building: "$address.building",
          coord: "$address.coord",
          street: "$address.street",
          zipcode: "$address.zipcode",
        },
        cuisine: 1,
        name: 1,
      },
  },
]
```

PIPELINE OUTPUT
Sample of 10 documents

```
▸ address: Object
  cuisine: "Bakery"
  name: "Morris Park Bake Shop"

▸ address: Object
  cuisine: "Hamburgers"
  name: "Wendy'S"

▸ address: Object
  cuisine: "Irish"
  name: "Dj Reynolds Pub And Restaurant"

▸ address: Object
  cuisine: "American "
  name: "Riviera Caterer"

▸ address: Object
  cuisine: "Jewish/Kosher"
  name: "Tov Kosher Kitchen"
```

2. Write a query to display the first 5 restaurant which is in the borough *Manhattan*.

```
[
  {
   $project:
     {
       _id: 0,
     },
  },
  {
   $match: {
     borough: "Manhattan",
    },
  },
  {
   $limit: 5,
  },
] },
]
```



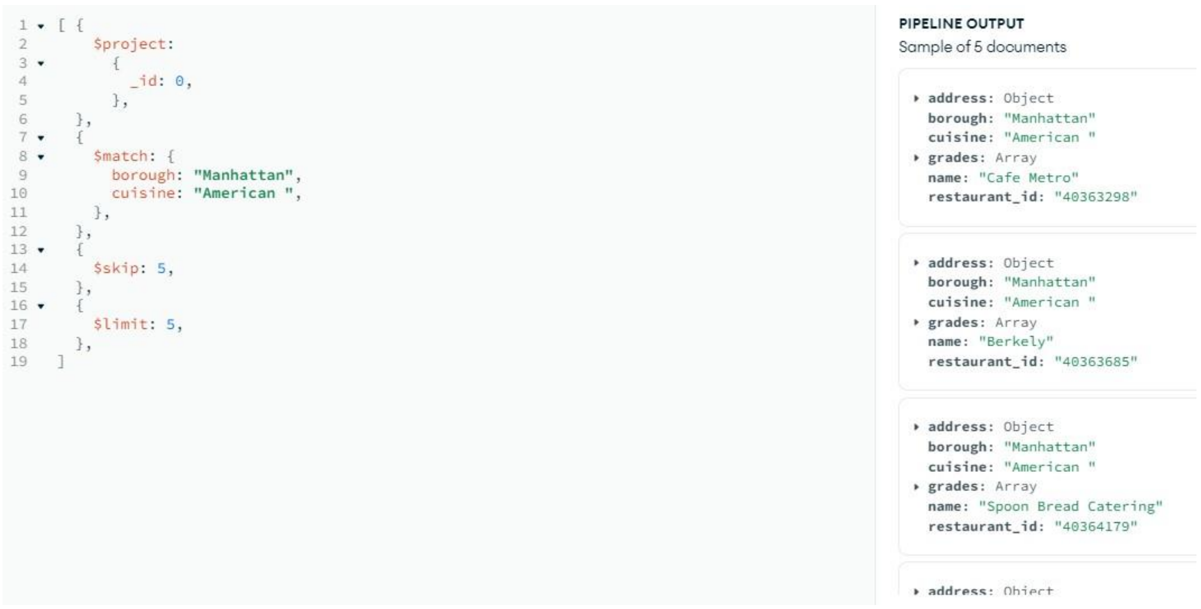
```
[
  {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match: {
      borough: "Manhattan",
    },
  },
  {
    $limit: 5,
  },
]
```

PIPELINE OUTPUT
Sample of 5 documents

```
▸ address: Object
  borough: "Manhattan"
  cuisine: "Irish"
▸ grades: Array
  name: "Dj Reynolds Pub And Restaurant"
  restaurant_id: "30191841"

▸ address: Object
  borough: "Manhattan"
  cuisine: "American "
▸ grades: Array
  name: "1 East 66Th Street Kitchen"
  restaurant_id: "40359480"
```

3. Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough *Manhattan* and serve cuisine *American*.

```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
   $match: {
     borough: "Manhattan",
     cuisine: "American ",
   },
  },
  {
   $skip: 5,
  },
  {
   $limit: 5,
  },
]
```



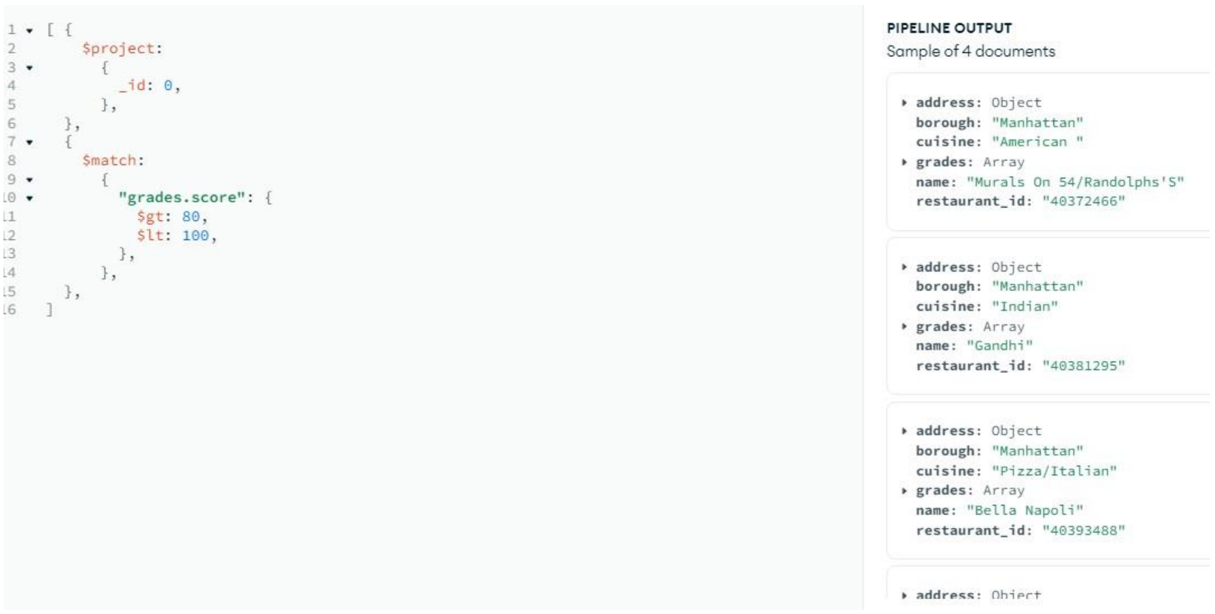
```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match: {
      borough: "Manhattan",
      cuisine: "American ",
    },
  },
  {
    $skip: 5,
  },
  {
    $limit: 5,
  },
]
```

PIPELINE OUTPUT
Sample of 5 documents

```
▸ address: Object
  borough: "Manhattan"
  cuisine: "American "
▸ grades: Array
  name: "Cafe Metro"
  restaurant_id: "40363298"

▸ address: Object
  borough: "Manhattan"
  cuisine: "American "
▸ grades: Array
  name: "Berkely"
  restaurant_id: "40363685"

▸ address: Object
  borough: "Manhattan"
  cuisine: "American "
▸ grades: Array
  name: "Spoon Bread Catering"
  restaurant_id: "40364179"

▸ address: Object
```

4. Write a query to find the restaurants that achieved a score, more than 80 but less than 100.

```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match:
      {
        "grades.score": {
          $gt: 80,
          $lt: 100,
        },
      },
  },
]
```



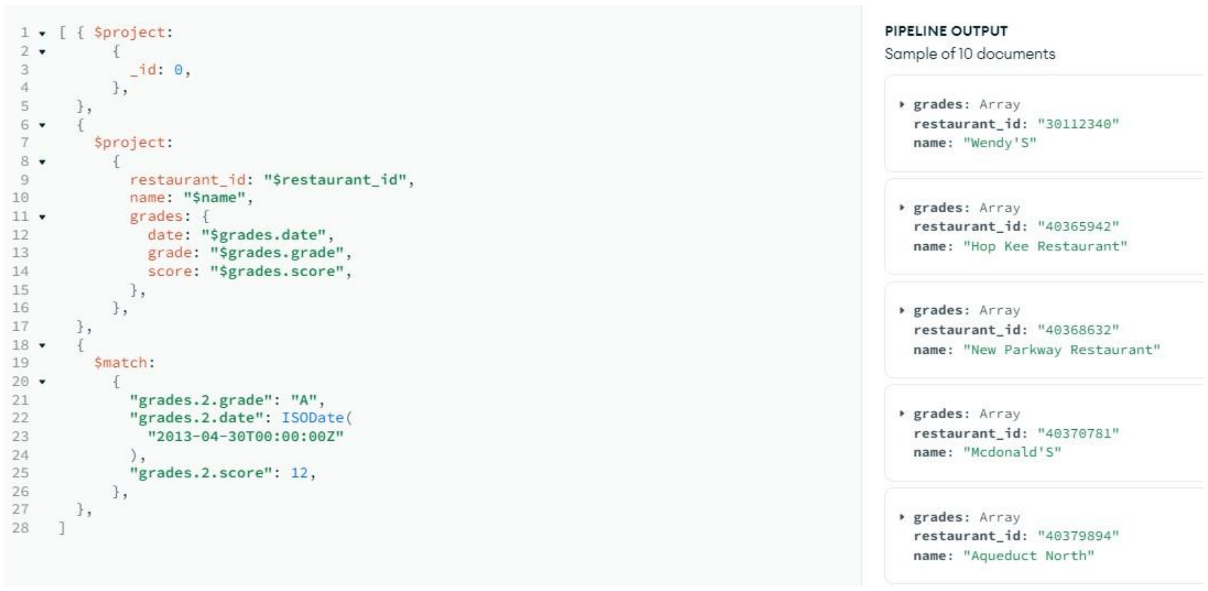
```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match:
      {
        "grades.score": {
          $gt: 80,
          $lt: 100,
        },
      },
  },
]
```

PIPELINE OUTPUT
Sample of 4 documents

```
▸ address: Object
  borough: "Manhattan"
  cuisine: "American "
▸ grades: Array
  name: "Murals On 54/Randolphs'S"
  restaurant_id: "40372466"

▸ address: Object
  borough: "Manhattan"
  cuisine: "Indian"
▸ grades: Array
  name: "Gandhi"
  restaurant_id: "40381295"

▸ address: Object
  borough: "Manhattan"
  cuisine: "Pizza/Italian"
▸ grades: Array
  name: "Bella Napoli"
  restaurant_id: "40393488"

▸ address: Object
```

5. Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 3rd element of grades array contains a grade of "A" and score 12 on an ISODate "2013-04-30T00:00:00Z".

```
[ { $project:
      {
        _id: 0,
      },
  },
  {
    $project:
      {
        restaurant_id: "$restaurant_id",
        name: "$name",
        grades: {
          date: "$grades.date",
          grade: "$grades.grade",
          score: "$grades.score",
        },
      },
  },
  {
    $match:
      {
        "grades.2.grade": "A",
        "grades.2.date": ISODate(
        "2013-04-30T00:00:00Z"
        ),
        "grades.2.score": 12,
      },
  },
]
```

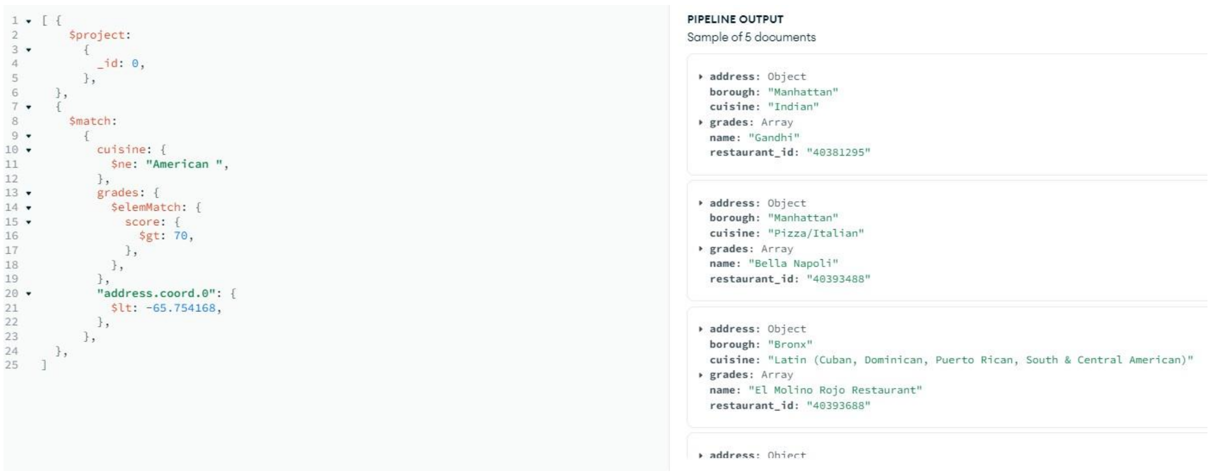


```
[ { $project:
      {
        _id: 0,
      },
  },
  {
    $project:
      {
        restaurant_id: "$restaurant_id",
        name: "$name",
        grades: {
          date: "$grades.date",
          grade: "$grades.grade",
          score: "$grades.score",
        },
      },
  },
  {
    $match:
      {
        "grades.2.grade": "A",
        "grades.2.date": ISODate(
          "2013-04-30T00:00:00Z"
        ),
        "grades.2.score": 12,
      },
  },
]
```

```
PIPELINE OUTPUT
Sample of 10 documents

▸ grades: Array
  restaurant_id: "30112340"
  name: "Wendy'S"

▸ grades: Array
  restaurant_id: "40365942"
  name: "Hop Kee Restaurant"

▸ grades: Array
  restaurant_id: "40368632"
  name: "New Parkway Restaurant"

▸ grades: Array
  restaurant_id: "40370781"
  name: "Mcdonald'S"

▸ grades: Array
  restaurant_id: "40379894"
  name: "Aqueduct North"
```

6. Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.

```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match:
      {
        cuisine: {
          $ne: "American ",
        },
        grades: {
          $elemMatch: {
            score: {
              $gt: 70,
            },
          },
        },
        "address.coord.0": {
          $lt: -65.754168,
        },
      },
  },
]
```



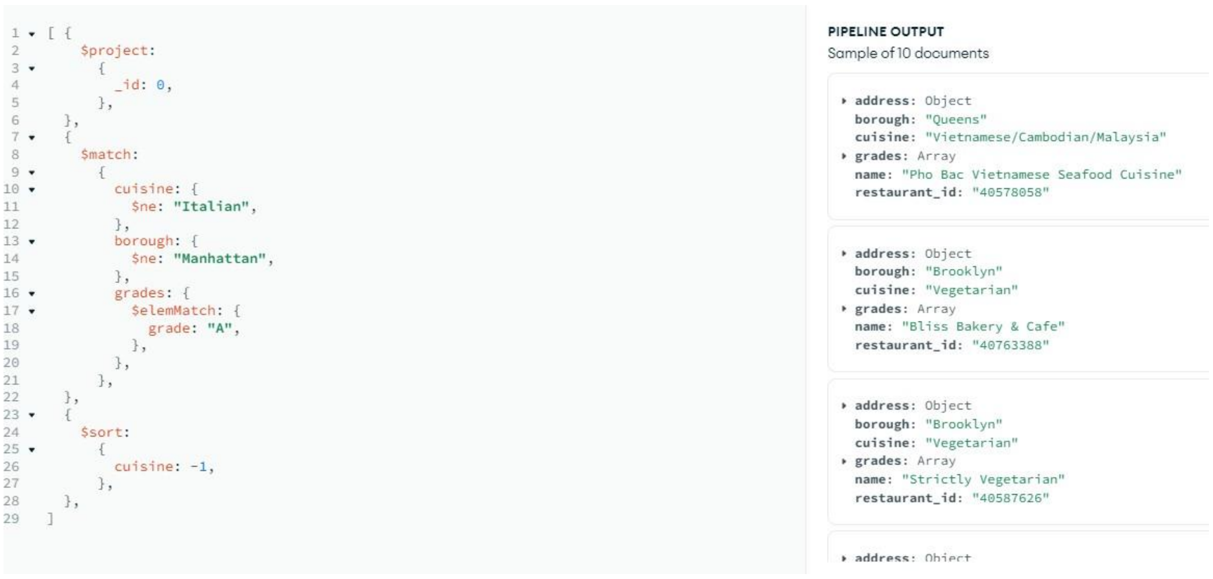
```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match:
      {
        cuisine: {
          $ne: "American ",
        },
        grades: {
          $elemMatch: {
            score: {
              $gt: 70,
            },
          },
        },
        "address.coord.0": {
          $lt: -65.754168,
        },
      },
  },
]
```

PIPELINE OUTPUT
Sample of 5 documents

```
▸ address: Object
  borough: "Manhattan"
  cuisine: "Indian"
▸ grades: Array
  name: "Gandhi"
  restaurant_id: "40381295"

▸ address: Object
  borough: "Manhattan"
  cuisine: "Pizza/Italian"
▸ grades: Array
  name: "Bella Napoli"
  restaurant_id: "40393488"

▸ address: Object
  borough: "Bronx"
  cuisine: "Latin (Cuban, Dominican, Puerto Rican, South & Central American)"
▸ grades: Array
  name: "El Molino Rojo Restaurant"
  restaurant_id: "40393688"

▸ address: Object
```

7. Write a query to find the restaurants which do not prepare any *Italian* cuisine and achieved a grade point 'A' not belongs to the borough Manhattan. The document must be displayed according to the cuisine in descending order.

```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match:
      {
        cuisine: {
          $ne: "Italian",
        },
        borough: {
          $ne: "Manhattan",
        },
        grades: {
          $elemMatch: {
            grade: "A",
          },
        },
      },
  },
  {
    $sort:
      {
        cuisine: -1,
      },
  },
]
```



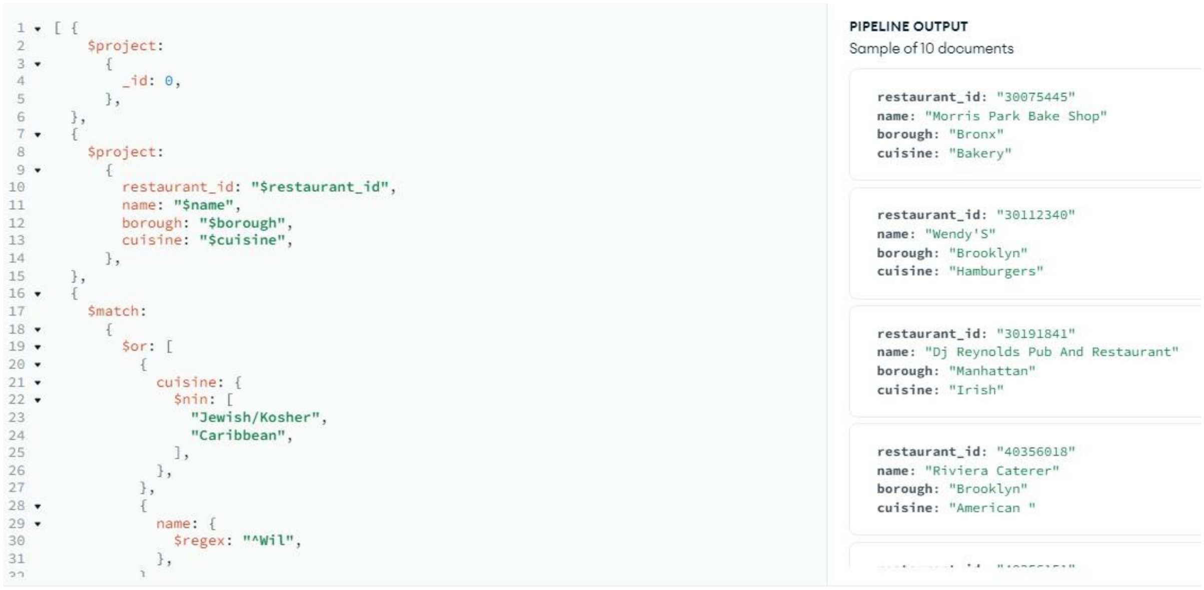
```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $match:
      {
        cuisine: {
          $ne: "Italian",
        },
        borough: {
          $ne: "Manhattan",
        },
        grades: {
          $elemMatch: {
            grade: "A",
          },
        },
      },
  },
  {
    $sort:
      {
        cuisine: -1,
      },
  },
]
```

PIPELINE OUTPUT
Sample of 10 documents

```
▸ address: Object
  borough: "Queens"
  cuisine: "Vietnamese/Cambodian/Malaysia"
▸ grades: Array
  name: "Pho Bac Vietnamese Seafood Cuisine"
  restaurant_id: "40578058"

▸ address: Object
  borough: "Brooklyn"
  cuisine: "Vegetarian"
▸ grades: Array
  name: "Bliss Bakery & Cafe"
  restaurant_id: "40763388"

▸ address: Object
  borough: "Brooklyn"
  cuisine: "Vegetarian"
▸ grades: Array
  name: "Strictly Vegetarian"
  restaurant_id: "40587626"

▸ address: Object
```

8. Write a query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'Jewish/Kosher' and 'Caribbean' or restaurant's name begins with letter 'Wil'.

```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $project:
      {
        restaurant_id: "$restaurant_id",
        name: "$name",
        borough: "$borough",
        cuisine: "$cuisine",
      },
  },
  {
    $match:
      {
        $or: [
          {
            cuisine: {
              $nin: [
                "Jewish/Kosher",
                "Caribbean",
              ],
            },
          },
          {
            name: {
              $regex: "^Wil",
            },
          },
        ],
      },
    },
  ]
```

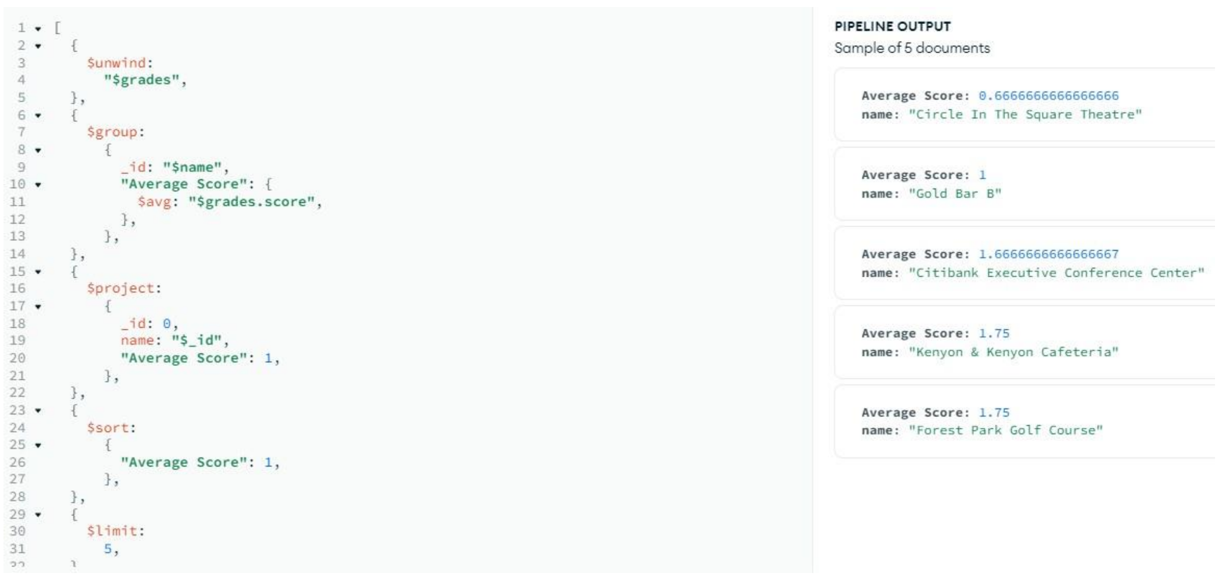


```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $project:
      {
        restaurant_id: "$restaurant_id",
        name: "$name",
        borough: "$borough",
        cuisine: "$cuisine",
      },
  },
  {
    $match:
      {
        $or: [
          {
            cuisine: {
              $nin: [
                "Jewish/Kosher",
                "Caribbean",
              ],
            },
          },
          {
            name: {
              $regex: "^Wil",
            },
```

```
PIPELINE OUTPUT
Sample of 10 documents

restaurant_id: "30075445"
name: "Morris Park Bake Shop"
borough: "Bronx"
cuisine: "Bakery"

restaurant_id: "30112340"
name: "Wendy'S"
borough: "Brooklyn"
cuisine: "Hamburgers"

restaurant_id: "30191841"
name: "Dj Reynolds Pub And Restaurant"
borough: "Manhattan"
cuisine: "Irish"

restaurant_id: "40356018"
name: "Riviera Caterer"
borough: "Brooklyn"
cuisine: "American "
```

9. Write an aggregate query to find average score obtained by each of restaurant. Sort the score in ascending order and only view the first 5 restaurant. (Hint : use *$unwind* to reconstruct the *grades* array).

```
[
  {
    $unwind:
      "$grades",
  },
  {
    $group:
      {
        _id: "$name",
        "Average Score": {
          $avg: "$grades.score",
        },
      },
  },
  {
    $project:
      {
        _id: 0,
        name: "$_id",
        "Average Score": 1,
      },
    },
    {
     $sort:
       {
        "Average Score": 1,
       },
    },
    {
     $limit:
       5,
    },
  ]
```



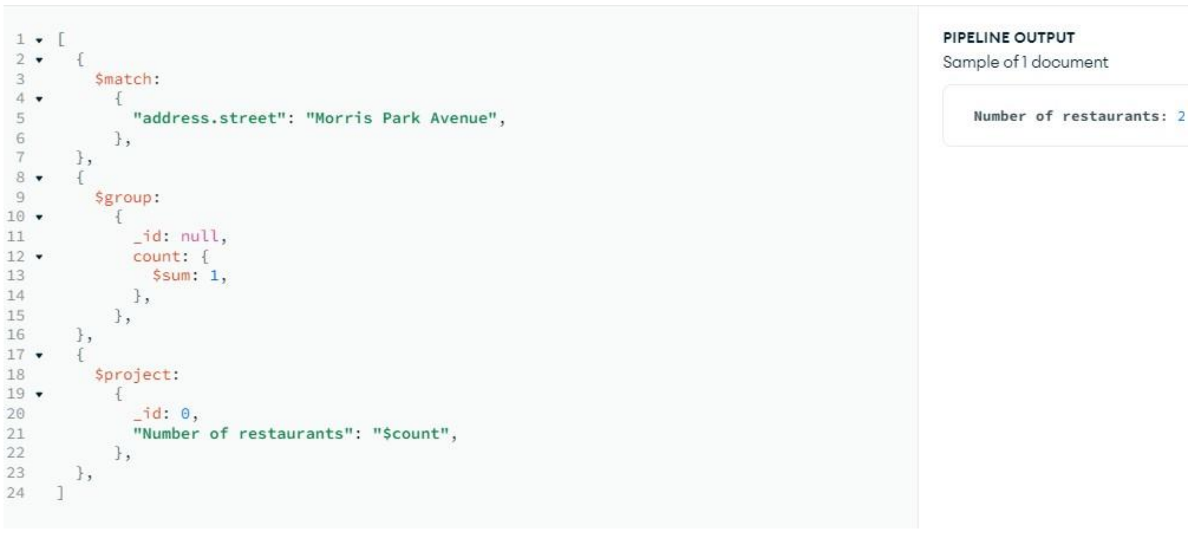
```
[
  {
    $unwind:
      "$grades",
  },
  {
    $group:
      {
        _id: "$name",
        "Average Score": {
          $avg: "$grades.score",
        },
      },
  },
  {
    $project:
      {
        _id: 0,
        name: "$_id",
        "Average Score": 1,
      },
  },
  {
    $sort:
      {
        "Average Score": 1,
      },
  },
  {
    $limit:
      5,
```

PIPELINE OUTPUT
Sample of 5 documents

```
Average Score: 0.6666666666666666
name: "Circle In The Square Theatre"

Average Score: 1
name: "Gold Bar B"

Average Score: 1.6666666666666667
name: "Citibank Executive Conference Center"

Average Score: 1.75
name: "Kenyon & Kenyon Cafeteria"

Average Score: 1.75
name: "Forest Park Golf Course"
```

10. Write a query to count the number of restaurants at the *Morris Park Avenue*.

```
  [
    {
     $match:
       {
        "address.street": "Morris Park Avenue",
       },
    },
    {
     $group:
       {
        _id: null,
        count: {
          $sum: 1,
        },
       },
    },
    {
     $project:
      {
        _id: 0,
        "Number of restaurants": "$count",
      },
    },
  ]
```



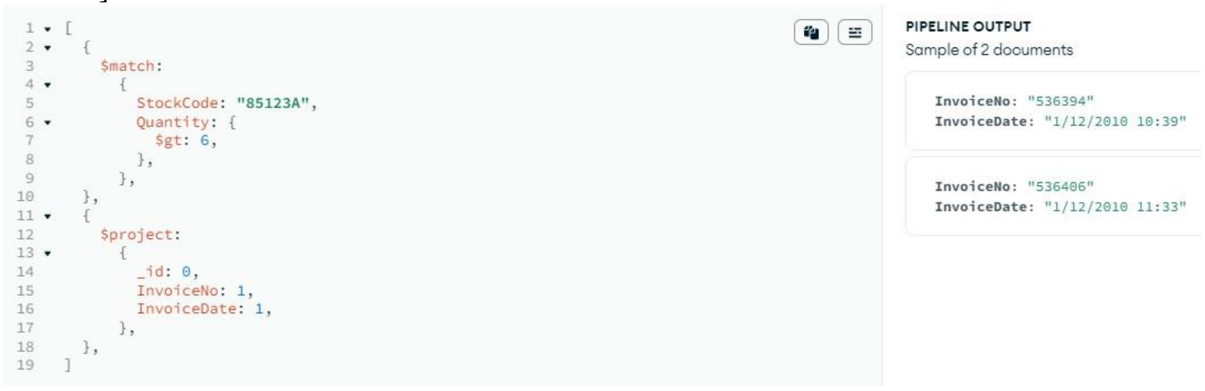
```
[
  {
    $match:
      {
        "address.street": "Morris Park Avenue",
      },
  },
  {
    $group:
      {
        _id: null,
        count: {
          $sum: 1,
        },
      },
  },
  {
    $project:
      {
        _id: 0,
        "Number of restaurants": "$count",
      },
  },
]
```

PIPELINE OUTPUT
Sample of 1 document

```
Number of restaurants: 2
```

1. Write query to display invoice number, invoice date for *StockCode* 85123A that have quantity order more than 6.

```
[
  {
    $match:
      {
        StockCode: "85123A",
        Quantity: {
          $gt: 6,
        },
      },
  },
  {
    $project:
      {
        _id: 0,
        InvoiceNo: 1,
        InvoiceDate: 1,
      },
  },
]
```



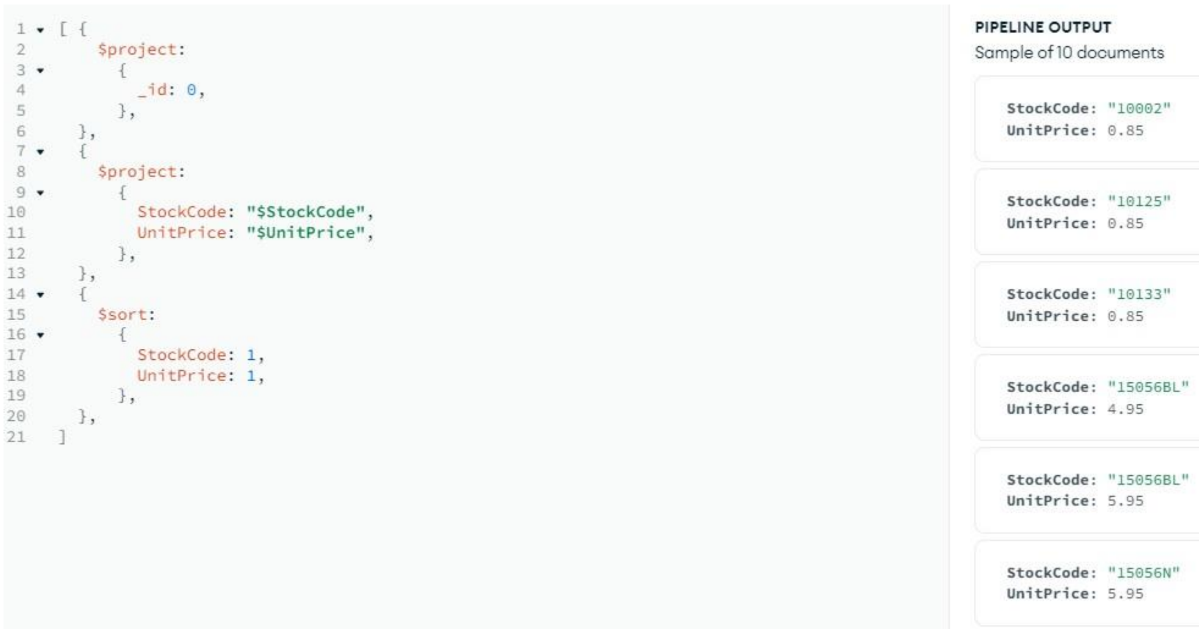
```
[
  {
    $match:
      {
        StockCode: "85123A",
        Quantity: {
          $gt: 6,
        },
      },
  },
  {
    $project:
      {
        _id: 0,
        InvoiceNo: 1,
        InvoiceDate: 1,
      },
  },
]
```

PIPELINE OUTPUT
Sample of 2 documents

```
InvoiceNo: "536394"
InvoiceDate: "1/12/2010 10:39"

InvoiceNo: "536406"
InvoiceDate: "1/12/2010 11:33"
```

2. Write a query to display only *StockCode* and *UnitPrice* (in ascending order).

```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $project:
      {
        StockCode: "$StockCode",
        UnitPrice: "$UnitPrice",
      },
  },
  {
    $sort:
      {
        StockCode: 1,
        UnitPrice: 1,
      },
  },
]
```



```
[ {
    $project:
      {
        _id: 0,
      },
  },
  {
    $project:
      {
        StockCode: "$StockCode",
        UnitPrice: "$UnitPrice",
      },
  },
  {
    $sort:
      {
        StockCode: 1,
        UnitPrice: 1,
      },
  },
]
```

PIPELINE OUTPUT

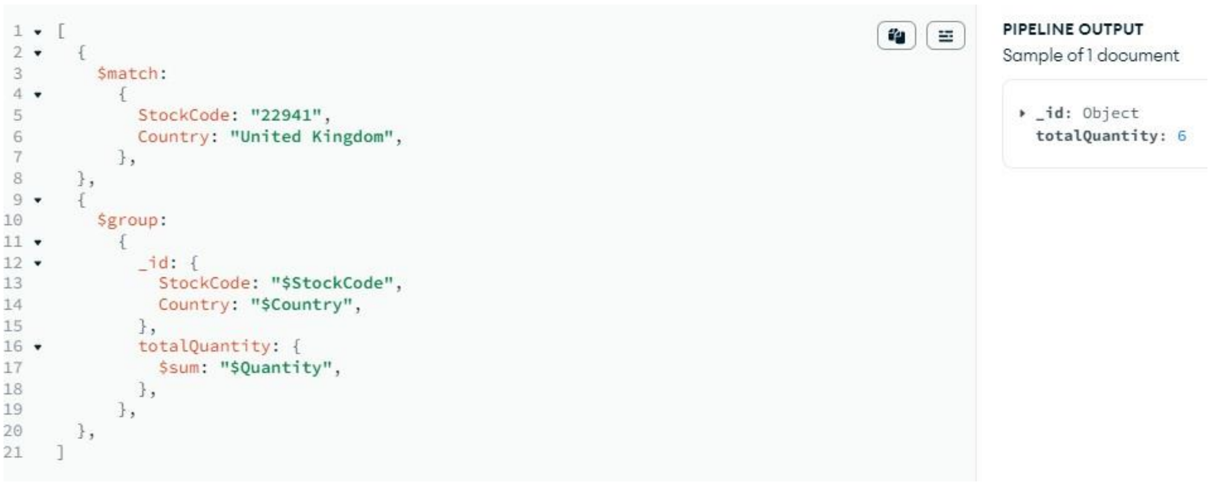
Sample of 10 documents

```
StockCode: "10002"
UnitPrice: 0.85

StockCode: "10125"
UnitPrice: 0.85

StockCode: "10133"
UnitPrice: 0.85

StockCode: "15056BL"
UnitPrice: 4.95

StockCode: "15056BL"
UnitPrice: 5.95

StockCode: "15056N"
UnitPrice: 5.95
```

3. Find the total quantity order for *StockCode* 22941 for customer in United Kingdom.

```
[
  {
    $match:
      {
        StockCode: "22941",
        Country: "United Kingdom",
      },
  },
  {
    $group:
      {
        _id: {
```

```
        StockCode: "$StockCode",
        Country: "$Country",
      },
      totalQuantity: {
        $sum: "$Quantity",
      },
    },
  },
]
```

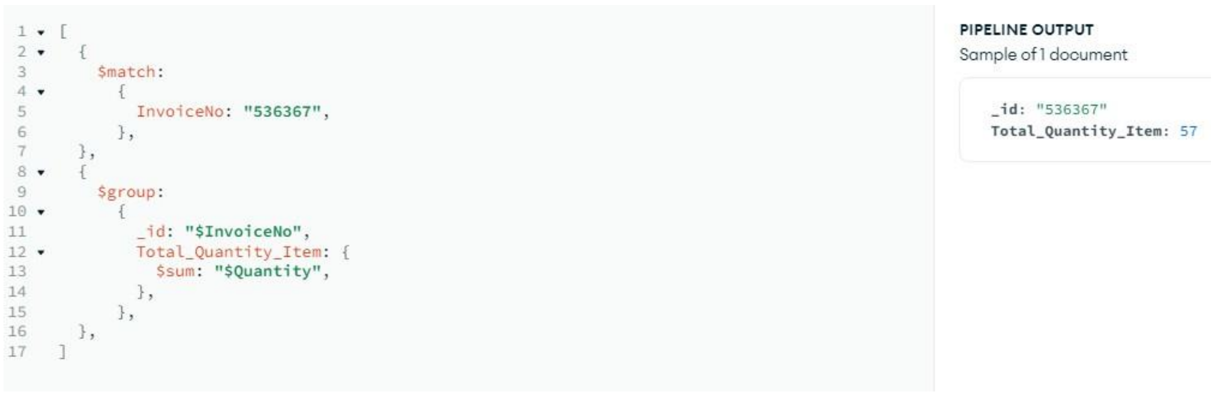


```
[
  {
    $match:
      {
        StockCode: "22941",
        Country: "United Kingdom",
      },
  },
  {
    $group:
      {
        _id: {
          StockCode: "$StockCode",
          Country: "$Country",
        },
        totalQuantity: {
          $sum: "$Quantity",
        },
      },
  },
]
```

```
▸ _id: Object
  totalQuantity: 6
```

4. Find the total quantity item purchase in invoice no 536367.

```
[
  {
    $match:
      {
        InvoiceNo: "536367",
      },
  },
  {
    $group:
      {
        _id: "$InvoiceNo",
        Total_Quantity_Item: {
          $sum: "$Quantity",
        },
      },
  },
]
```



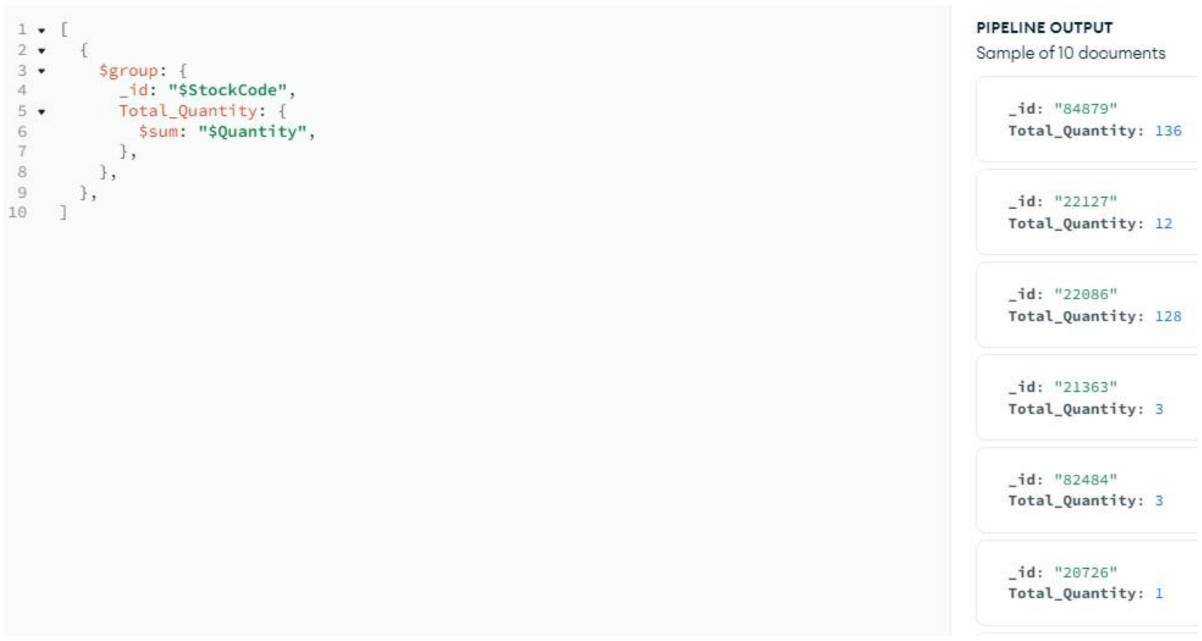
```
[
  {
    $match:
      {
        InvoiceNo: "536367",
      },
  },
  {
    $group:
      {
        _id: "$InvoiceNo",
        Total_Quantity_Item: {
          $sum: "$Quantity",
        },
      },
  },
]
```

PIPELINE OUTPUT
Sample of 1 document

```
_id: "536367"
Total_Quantity_Item: 57
```

5. Find the total quantity order for each *StockCode*.

```
[
  {
   $group:
     {
      _id: "$StockCode",
      Total_Quantity: {
        $sum: "$Quantity",
      },
     },
  },
]
```



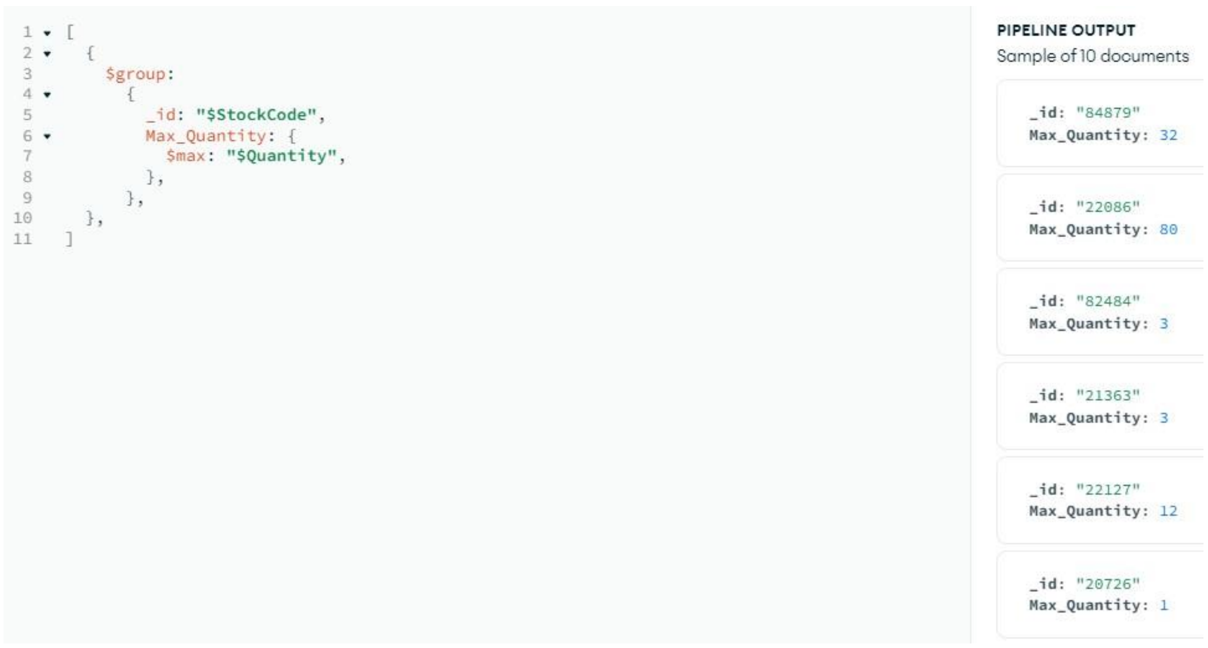
```
[
  {
    $group: {
      _id: "$StockCode",
      Total_Quantity: {
        $sum: "$Quantity",
      },
    },
  },
]
```

PIPELINE OUTPUT
Sample of 10 documents

```
_id: "84879"
Total_Quantity: 136

_id: "22127"
Total_Quantity: 12

_id: "22086"
Total_Quantity: 128

_id: "21363"
Total_Quantity: 3

_id: "82484"
Total_Quantity: 3

_id: "20726"
Total_Quantity: 1
```

6. Find the maximum quantity order of each *StockCode*.

```
[
  {
    $group:
      {
        _id: "$StockCode",
        Max_Quantity: {
          $max: "$Quantity",
        },
      },
  },
]
```



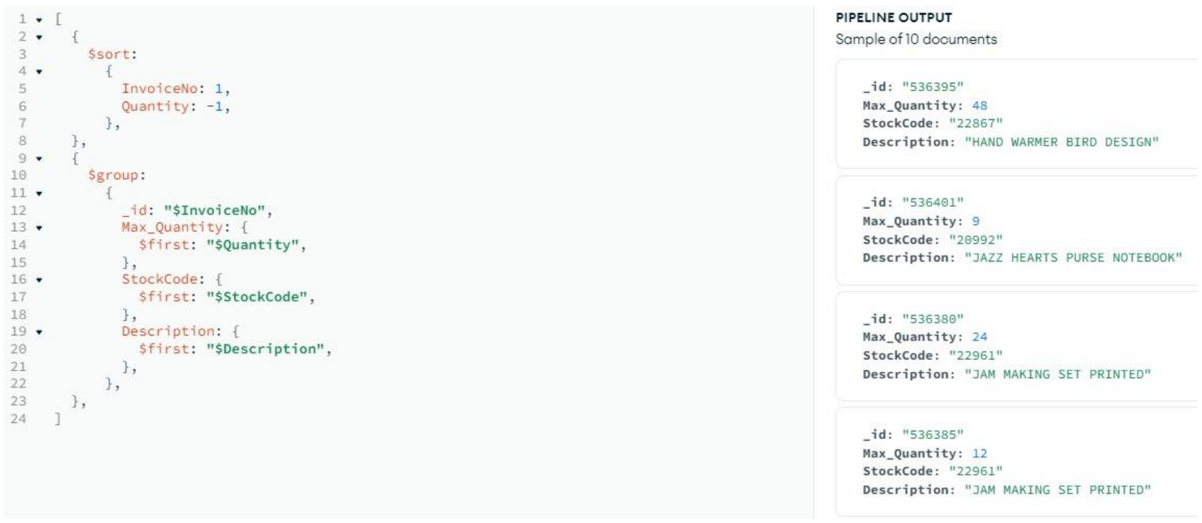
```
[
  {
    $group:
      {
        _id: "$StockCode",
        Max_Quantity: {
          $max: "$Quantity",
        },
      },
  },
]
```

PIPELINE OUTPUT
Sample of 10 documents

```
_id: "84879"
Max_Quantity: 32

_id: "22086"
Max_Quantity: 80

_id: "82484"
Max_Quantity: 3

_id: "21363"
Max_Quantity: 3

_id: "22127"
Max_Quantity: 12

_id: "20726"
Max_Quantity: 1
```

7. Find the StockCode and Description of maximum quantity in each order.

```
[
  {
    $sort:
      {
        InvoiceNo: 1,
        Quantity: -1,
      },
  },
  {
    $group:
      {
        _id: "$InvoiceNo",
        Max_Quantity: {
          $first: "$Quantity",
        },
        StockCode: {
          $first: "$StockCode",
        },
        Description: {
          $first: "$Description",
        },
      },
    },
  ]
```



```
[
  {
    $sort:
      {
        InvoiceNo: 1,
        Quantity: -1,
      },
  },
  {
    $group:
      {
        _id: "$InvoiceNo",
        Max_Quantity: {
          $first: "$Quantity",
        },
        StockCode: {
          $first: "$StockCode",
        },
        Description: {
          $first: "$Description",
        },
      },
  },
]
```

PIPELINE OUTPUT

Sample of 10 documents

```
_id: "536395"
Max_Quantity: 48
StockCode: "22867"
Description: "HAND WARMER BIRD DESIGN"

_id: "536401"
Max_Quantity: 9
StockCode: "20992"
Description: "JAZZ HEARTS PURSE NOTEBOOK"

_id: "536380"
Max_Quantity: 24
StockCode: "22961"
Description: "JAM MAKING SET PRINTED"

_id: "536385"
Max_Quantity: 12
StockCode: "22961"
Description: "JAM MAKING SET PRINTED"
```