Restaurant Queries MongoDB

1. Write a MongoDB query to display all the documents in the collection restaurants.

```
db.restaurants.find();
```

2. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all the documents in the collection restaurant.

```
db.restaurants.find({},{"restaurant_id" : 1,"name":1,"borough":1,"cuisine" :1});
```

3. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine, but exclude the field _id for all the documents in the collection restaurant.

```
db.restaurants.find({},{"restaurant_id" : 1,"name":1,"borough":1,"cuisine" :1,"_id":0});
```

4. Write a MongoDB query to display the fields restaurant_id, name, borough and zip code, but exclude the field _id for all the documents in the collection restaurant.

```
db.restaurants.find({},{"restaurant_id" :1,"name":1,"borough":1,"address.zipcode" :1,"_id":0});
```

5. Write a MongoDB query to display all the restaurant which is in the borough Bronx.

```
db.restaurants.find({"borough": "Bronx"});
```

6. Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx.

```
db.restaurants.find({"borough": "Bronx"}).limit(5);
```

7. Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.

```
db.restaurants.find({"borough": "Bronx"}).skip(5).limit(5);
```

8. Write a MongoDB query to find the restaurants who achieved a score more than 90.

```
db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 90}}}});
```

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

```
db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 80 , $lt :100}}}});
```

10. Write a MongoDB query to find the restaurants which locate in latitude value less than -95.754168.

```
db.restaurants.find({"address.coord" : {$lt : -95.754168}});
```

11. 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.

```
db.restaurants.find(
        {$and:
           [
             {"cuisine" : {$ne :"American "}},
             {"grades.score" : {$gt : 70}},
             {"address.coord" : {$lt : -65.754168}}
           ]
         }
           );
```

12. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a score more than 70 and located in the longitude less than -65.754168.
Note : Do this query without using $and operator.

```
db.restaurants.find(
                    {
                     "cuisine" : {$ne : "American "},
                     "grades.score" :{$gt: 70},
                     "address.coord" : {$lt : -65.754168}
                    }
                );
```

13. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.

```
db.restaurants.find( {
                     "cuisine" : {$ne : "American"},
                     "grades.grade" :"A",
                     "borough": {$ne : "Brooklyn"}
                  }
               ).sort({"cuisine":-1});
```

14. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Wil' as first three letters for its name.

```
db.restaurants.find(
{name: /^Wil/},
{
"restaurant_id" : 1,
"name":1,"borough":1,
"cuisine" :1
}
);
```

15. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.

```
db.restaurants.find(
{name: /ces$/},
{
"restaurant_id" : 1,
"name":1,"borough":1,
"cuisine" :1
}
);
```

16. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.

```
db.restaurants.find(
{"name": /.*Reg.*/},
{
"restaurant_id" : 1,
"name":1,"borough":1,
"cuisine" :1
```

```
}
);
```

17. Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.

```
db.restaurants.find(
{
"borough": "Bronx" ,
$or : [
{ "cuisine" : "American " },
{ "cuisine" : "Chinese" }
]
}
);
```

18. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronx or Brooklyn.

```
db.restaurants.find(
{"borough" :{$in :["Staten Island","Queens","Bronx","Brooklyn"]}},
{
"restaurant_id" : 1,
"name":1,"borough":1,
"cuisine" :1
}
);
```

19. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.

```
db.restaurants.find(
{"borough" :{$nin :["Staten Island","Queens","Bronx","Brooklyn"]}},
{
"restaurant_id" : 1,
"name":1,"borough":1,
"cuisine" :1
}
);
```

20. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10.

```
db.restaurants.find(
{"grades.score" :
{ $not:
{$gt : 10}
}
},
{
"restaurant_id" : 1,
"name":1,"borough":1,
"cuisine" :1
}
);
```

21. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

```
db.restaurants.find(
{$or: [
 {name: /^Wil/},
 {"$and": [
    {"cuisine" : {$ne :"American "}},
    {"cuisine" : {$ne :"Chinees"}}
  ]}
]}
,{"restaurant_id" : 1,"name":1,"borough":1,"cuisine" :1}
);
```

22. Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates.

```
db.restaurants.find(
          {
           "grades.date": ISODate("2014-08-11T00:00:00Z"),
           "grades.grade":"A" ,
           "grades.score" : 11
          },
          {"restaurant_id" : 1,"name":1,"grades":1}
        );
```

23. Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

```
db.restaurants.find(
              { "grades.1.date": ISODate("2014-08-11T00:00:00Z"),
                "grades.1.grade":"A" ,
                "grades.1.score" : 9
              },
               {"restaurant_id" : 1,"name":1,"grades":1}
            );
```

24. Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52.

```
db.restaurants.find(
              {
                "address.coord.1": {$gt : 42, $lte : 52}
              },
               {"restaurant_id" : 1,"name":1,"address":1,"coord":1}
            );
```

25. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.

```
db.restaurants.find().sort({"name":1});
```

26. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.

```
db.restaurants.find().sort(
                {"name":-1}
                );
```

27. Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.

```
db.restaurants.find().sort(
                 {"cuisine":1,"borough" : -1,}
                );
```

28. Write a MongoDB query to know whether all the addresses contains the street or not.

```
db.restaurants.find(
              {"address.street" :
                 { $exists : true }
              }
             );
```

29. Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is Double.

```
db.restaurants.find(
              {"address.coord" :
                {$type : 1}
              }
             );
```

30. Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

```
db.restaurants.find(
               {"grades.score" :
                 {$mod : [7,0]}
               },
                 {"restaurant_id" : 1,"name":1,"grades":1}
              );
```

31. Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.

```
db.restaurants.find(
             { name :
               { $regex : "mon.*", $options: "i" }
             },
                {
                  "name":1,
                  "borough":1,
                  "address.coord":1,
                  "cuisine" :1
```

```
    }
);
```

32. Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.

```
db.restaurants.find(
    { name :
      { $regex : /^Mad/i, }
    },
       {
        "name":1,
        "borough":1,
        "address.coord":1,
        "cuisine" :1
       }
);
```

33. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5.

```
db.restaurants.find({ "grades.score": { $lt: 5 } })
```

34. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan.

```
db.restaurants.find({ "grades.score": { $lt: 5 }, "borough": "Manhattan" })
```

35. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn.

```
db.restaurants.find({
  $and: [
    {
      $or: [
        {borough: "Manhattan"},
        {borough: "Brooklyn"}
      ]
    },
    {
      "grades.score": { $lt: 5 }
    }
  ]
})
```

36. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

```
db.restaurants.find({
  $and: [
{ $or: [{ borough: "Manhattan" }, { borough: "Brooklyn" }] },
{ "grades.score": { $lt: 5 } },
{ cuisine: { $ne: "American" } }
  ]
})
```

37. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

```
db.restaurants.find({
   $and: [
      {
         $or: [
            {borough: "Manhattan"},
            {borough: "Brooklyn"}
         ]
      },
      {
         $nor: [
            {cuisine: "American"},
            {cuisine: "Chinese"}
         ]
      },
      {
grades: {
            $elemMatch: {
score: { $lt: 5 }
            }
         }
      }
   ]
})
```

38. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6.

```
db.restaurants.find({
  $and: [
   {"grades.score": 2},
   {"grades.score": 6}
  ]
})
```

39. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan.

```
db.restaurants.find({
  $and: [
   {"grades.score": 2},
   {"grades.score": 6},
   {"borough": "Manhattan"}
  ]
})
```

40. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn.

```
db.restaurants.find({
  $and: [
   {"grades.score": 2},
   {"grades.score": 6},
   {"borough": {"$in": ["Manhattan", "Brooklyn"]}}
  ]
})
```

41. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

```
db.restaurants.find({
   $and: [
      {borough: {$in: ["Manhattan", "Brooklyn"]}},
      {"grades.score": {$all: [2, 6]}},
      {cuisine: {$ne: "American"}}
   ]
})
```

42. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

```
db.restaurants.find({
  $and: [
{ borough: { $in: ["Manhattan", "Brooklyn"] } },
{ cuisine: { $nin: ["American", "Chinese"] } },
{ grades: { $elemMatch: { score: 2 } } },
{ grades: { $elemMatch: { score: 6 } } }
  ]
})
```

43. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6.

```
db.restaurants.find({
   $or: [
{ "grades.score": 2 },
{ "grades.score": 6 }
   ]
})
```

44. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan.

```
db.restaurants.find({
   $and: [
      {
         $or: [
{ "grades.score": 2 },
{ "grades.score": 6 }
         ]
```

```
      },
{ "borough": "Manhattan" }
   ]
})
```

45. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn.

```
db.restaurants.find({
  $and: [
    {
      $or: [
{ borough: "Manhattan" },
{ borough: "Brooklyn" }
      ]
    },
    {
      $or: [
{ "grades.score": 2 },
{ "grades.score": 6 }
      ]
    }
  ]
})
```

46. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

```
db.restaurants.find({
  $and: [
    {
      $or: [
{ borough: "Manhattan" },
{ borough: "Brooklyn" }
      ]
    },
    {
      $or: [
{ "grades.score": 2 },
{ "grades.score": 6 }
      ]
    },
    {
cuisine: { $ne: "American" }
    }
  ]
})
```

47. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

```
db.restaurants.find({
  $and: [
    {
      $or: [
{ "grades.score": 2 },
{ "grades.score": 6 }
      ]
    },
    {
      $or: [
{ borough: "Manhattan" },
{ borough: "Brooklyn" }
      ]
    },
    {
      $nor: [
{ cuisine: "American" },
{ cuisine: "Chinese" }
      ]
    }
  ]
})
```

48. Write a MongoDB query to find the restaurants that have all grades with a score greater than 5.

```
db.restaurants.find({
  "grades": {
    "$not": {
      "$elemMatch": {
        "score": {
          "$lte": 5
        }
      }
    }
  }
})
```

49. Write a MongoDB query to find the restaurants that have all grades with a score greater than 5 and are located in the borough of Manhattan.

```
db.restaurants.find({
  "borough": "Manhattan",
  "grades": {
    "$not": {
      "$elemMatch": {
        "score": {
          "$lte": 5
        }
      }
    }
```

```
  }
})
```

50. Write a MongoDB query to find the restaurants that have all grades with a score greater than 5 and are located in the borough of Manhattan or Brooklyn.

```
db.restaurants.find({
  "borough": {
    "$in": ["Manhattan", "Brooklyn"]
  },
  "grades": {
    "$not": {
      "$elemMatch": {
        "score": {
          "$lte": 5
        }
      }
    }
  }
})
```

51. Write a MongoDB query to find the average score for each restaurant.

```
db.restaurants.aggregate([{
    $unwind: "$grades"
  },
  {
    $group: {
      _id: "$name",
      avgScore: {
        $avg: "$grades.score"
      }
    }
  }
])
```

52. Write a MongoDB query to find the highest score for each restaurant.

```
db.restaurants.aggregate([{
    $unwind: "$grades"
  },
  {
    $group: {
      _id: "$name",
      highest_score: {
        $max: "$grades.score"
      }
    }
  }
])
```

53. Write a MongoDB query to find the lowest score for each restaurant.

```
db.restaurants.aggregate([{
   $unwind: "$grades"
  },
  {
   $group: {
     _id: "$name",
     lowest_score: {
       $min: "$grades.score"
     }
   }
  }
])
```

54. Write a MongoDB query to find the count of restaurants in each borough.

```
db.restaurants.aggregate([{
  $group: {
    _id: "$borough",
    count: {
      $sum: 1
    }
  }
}])
```

55. Write a MongoDB query to find the count of restaurants for each cuisine.

```
db.restaurants.aggregate([{
  $group: {
    _id: "$cuisine",
    count: {
      $sum: 1
    }
  }
}])
```

56. Write a MongoDB query to find the count of restaurants for each cuisine and borough.

```
db.restaurants.aggregate([{
  $group: {
    _id: {
      cuisine: "$cuisine",
      borough: "$borough"
    },
    count: {
      $sum: 1
    }
  }
}])
```

57. Write a MongoDB query to find the count of restaurants that received a grade of 'A' for each cuisine.

```
db.restaurants.aggregate([
 {
  $unwind: "$grades"
 },
 {
  $match: { "grades.grade": "A" }
 },
 {
  $group: {
   _id: "$cuisine",
count: { $sum: 1 }
  }
 }
])
```

58. Write a MongoDB query to find the count of restaurants that received a grade of 'A' for each borough.

```
db.restaurants.aggregate([
 {
  $unwind: "$grades"
 },
 {
  $match: { "grades.grade": "A" }
 },
 {
  $group: {
   _id: "$borough",
count: { $sum: 1 }
  }
 }
])
```

5**9. Write a MongoDB query to find the count of restaurants that received a grade of 'A' for each cuisine and borough.**

```
db.restaurants.aggregate([
 {
  $match: { "grades.grade": "A" }
 },
 {
  $group: {
   _id: { cuisine: "$cuisine", borough: "$borough" },
count: { $sum: 1 }
  }
 },
 {
  $sort: { count: -1 }
 }
]);
```

60. Write a MongoDB query to find the number of restaurants that have been graded in each month of the year.

```
db.restaurants.aggregate([
 {
  $unwind: "$grades"
 },
 {
  $project: {
month: { $month: { $toDate: "$grades.date" } },
year: { $year: { $toDate: "$grades.date" } }
  }
 },
 {
  $group: {
   _id: { month: "$month", year: "$year" },
count: { $sum: 1 }
  }
 },
 {
  $sort: {
   "_id.year": 1,
   "_id.month": 1
  }
 }
]);
```

61. Write a MongoDB query to find the average score for each cuisine.

```
db.restaurants.aggregate([
 {
  $unwind: "$grades"
 },
 {
  $group: {
   _id: "$cuisine",
avgScore: { $avg: "$grades.score" }
  }
 }
])
```

62. Write a MongoDB query to find the highest score for each cuisine.

```
db.restaurants.aggregate([
 {
  $unwind: "$grades"
 },
 {
  $group: {
   _id: "$cuisine",
maxScore: { $max: "$grades.score" }
  }
 }
])
```

63. Write a MongoDB query to find the lowest score for each cuisine.

```
db.restaurants.aggregate([
 {
  $unwind: "$grades"
 },
 {
  $group: {
   _id: "$cuisine",
minScore: { $min: "$grades.score" }
  }
 }
])
```

64. Write a MongoDB query to find the average score for each borough.

```
db.restaurants.aggregate([
{ $unwind: "$grades" },
{ $group: { _id: "$borough", avgScore: { $avg: "$grades.score" } } }
])
```

65. Write a MongoDB query to find the highest score for each borough.

```
db.restaurants.aggregate([
{ $unwind: "$grades" },
{ $group: {
    _id: { borough: "$borough" },
highestScore: { $max: "$grades.score" }
  }
 }
])
```

66. Write a MongoDB query to find the lowest score for each borough.

```
db.restaurants.aggregate([
{ $unwind: "$grades" },
{ $group: {
    _id: { borough: "$borough" },
lowestScore: { $min: "$grades.score" }
  }
 }
])
```

67. Write a MongoDB query to find the name and address of the restaurants that received a grade of 'A' on a specific date.

```
db.restaurants.find(
 {
  "grades": {
   "$elemMatch": {
    "date": {
     "$eq": ISODate("2013-07-22T00:00:00Z")
    },
    "grade": {
     "$eq": "A"
```

```
      }
     }
    }
  },
  {
    "name": 1,
    "address": 1,
    "_id": 0
  }
)
```

68. Write a MongoDB query to find the name and address of the restaurants that received a grade of 'B' or 'C' on a specific date.

```
db.restaurants.find(
  {
    "grades": {
      $elemMatch: {
        "date": ISODate("2013-04-05"),
        "grade": { $in: [ "B", "C" ] }
      }
    }
  },
  {
    "name": 1,
    "address": 1
  }
)
```

69. Write a MongoDB query to find the name and address of the restaurants that have at least one 'A' grade and one 'B' grade.

```
db.restaurants.find({
  $and: [
{ "grades.grade": "A" },
{ "grades.grade": "B" }
  ]
},
{ name: 1, address: 1, _id: 0 })
```

70. Write a MongoDB query to find the name and address of the restaurants that have at least one 'A' grade and no 'B' grades.

```
Cdb.restaurants.find({
  $and: [
{ "grades.grade": "A" },
{ "grades.grade": { $not: { $eq: "B" } } }
  ]
},
{ name: 1, address: 1, _id: 0 })
```

71. Write a MongoDB query to find the name ,address and grades of the restaurants that have at least one 'A' grade and no 'C' grades.

```
db.restaurants.find({
  $and: [
{ "grades.grade": "A" },
{ "grades.grade": { $not: { $eq: "C" } } }
  ]
},
{ name: 1, address: 1, "grades.grade":1, _id: 0 })
```

72. Write a MongoDB query to find the name, address, and grades of the restaurants that have at least one 'A' grade, no 'B' grades, and no 'C' grades.

```
db.restaurants.find({
  $and: [
{ "grades.grade": "A" },
{ "grades.grade": { $not: { $eq: "B" } } },
{ "grades.grade": { $not: { $eq: "C" } } }
  ]
},
{ name: 1, address: 1, "grades.grade":1, _id: 0 })
```

73. Write a MongoDB query to find the name and address of the restaurants that have the word 'coffee' in their name.

```
db.restaurants.find({ name: { $regex: /coffee/i } }, { name: 1, address: 1 })
```

74. Write a MongoDB query to find the name and address of the restaurants that have a zipcode that starts with '10'.

```
db.restaurants.aggregate([
  {
    $unwind: "$address"
  },
  {
    $match: {
      "address.zipcode": /^10/
    }
  },
  {
    $project: {
name: 1,
      "address.street": 1,
      "address.zipcode": 1,
      _id: 0
    }
  }
])
```

75. Write a MongoDB query to find the name and address of the restaurants that have a cuisine that starts with the letter 'B'.

```
db.restaurants.find(
        { "cuisine": { $regex: /^B/ } },
        { "name": 1,
        "address": 1,
        "cuisine" : 1,
        "_id": 0 }
)
```

76. Write a MongoDB query to find the name, address, and cuisine of the restaurants that have a cuisine that ends with the letter 'y'.

```
db.restaurants.find(
{ cuisine: { $regex: /y$/i } },
{ name: 1,
address: 1,
cuisine: 1,
   _id: 0 }
)
```

77. Write a MongoDB query to find the name, address, and cuisine of the restaurants that have a cuisine that contains the word 'Pizza'.

```
db.restaurants.find(
{ cuisine: { $regex: /Pizza/i } },
{ name: 1, address: 1, cuisine: 1, _id: 0 }
)
```

78. Write a MongoDB query to find the restaurants achieved highest average score.

```
db.restaurants.aggregate([
 {$unwind: "$grades"},
 {$group: {
   _id: "$restaurant_id",
avgScore: {$avg: "$grades.score"}
 }},
 {$sort: {avgScore: -1}},
 {$limit: 1},
 {$project: {_id: 1, avgScore: 1}}
])
```

79. Write a MongoDB query to find all the restaurants with the highest number of "A" grades.

```
db.restaurants.aggregate([
 {$unwind: "$grades"},
 {$match: {"grades.grade": "A"}},
 {$group: {
   _id: "$restaurant_id",
count: {$sum: 1}
 }},
 {$sort: {count: -1}},
 {$group: {
   _id: "$count",
restaurants: {$push: "$_id"}
```

```
  }},
  {$sort: {_id: -1}},
  {$limit: 1},
  {$project: {restaurants: 1}}
])
```

80. Write a MongoDB query to find the cuisine type that is most likely to receive a "C" grade.

```
db.restaurants.aggregate([
  {$unwind: "$grades"},
  {$match: {"grades.grade": "C"}},
  {$group: {_id: "$cuisine", count: {$sum: 1}}},
  {$sort: {count: -1}}
])
```

81. Write a MongoDB query to find the restaurant that has the highest average score for thecuisine "Turkish".

```
db.restaurants.aggregate([
{ $match: { cuisine: "Turkish" } },
{ $unwind: "$grades" },
{ $group: {
    _id: "$name",
avgScore: { $avg: "$grades.score" }
  }},
{ $sort: { avgScore: -1 } }
])
```

82. Write a MongoDB query to find the restaurants that achieved the highest total score.

```
db.restaurants.aggregate([
{ $unwind: "$grades" },
{ $group: {
    _id: "$name",
totalScore: { $sum: "$grades.score" }
  }},
{ $sort: { totalScore: -1 } },
{ $group: {
    _id: "$totalScore",
restaurants: { $push: "$_id" }
  }},
{ $sort: { _id: -1 } },
{ $limit: 1 },
{ $unwind: "$restaurants" },
{ $group: {
    _id: "$_id",
restaurants: { $push: "$restaurants" }
  }}
])
```

83. Write a MongoDB query to find all the Chinese restaurants in Brooklyn.

```
db.restaurants.find({"borough": "Brooklyn", "cuisine": "Chinese"})
```

84. Write a MongoDB query to find the restaurant with the most recent grade date.

```
db.restaurants.aggregate([
{ $unwind: "$grades" },
{ $sort: { "grades.date": -1 } },
{ $limit: 1 },
{ $project: { name: 1, "grades.date": 1, _id: 0 } }
])
```

85. Write a MongoDB query to find the top 5 restaurants with the highest average score for each cuisine type, along with their average scores.

```
db.restaurants.aggregate([
 {$unwind: "$grades"},
 {$group: {
   _id: {cuisine: "$cuisine", restaurant_id: "$restaurant_id"},
avgScore: {$avg: "$grades.score"}
 }},
 {$sort: {
   "_id.cuisine": 1,
avgScore: -1
 }},
 {$group: {
   _id: "$_id.cuisine",
topRestaurants: {$push: {restaurant_id: "$_id.restaurant_id", avgScore: "$avgScore"}}
 }},
 {$project: {
   _id: 0,
cuisine: "$_id",
topRestaurants: {$slice: ["$topRestaurants", 5]}
 }}
])
```

86. Write a MongoDB query to find the top 5 restaurants in each borough with the highest number of "A" grades.

```
db.restaurants.aggregate([
 {$unwind: "$grades"},
 {$match: {"grades.grade": "A"}},
 {$group: {
   _id: {borough: "$borough", restaurant_id: "$restaurant_id"},
gradeCount: {$sum: 1}
 }},
 {$sort: {
   "_id.borough": 1,
gradeCount: -1
 }},
 {$group: {
   _id: "$_id.borough",
topRestaurants: {$push: {restaurant_id: "$_id.restaurant_id", gradeCount: "$gradeCount"}}
 }},
 {$project: {
   _id: 0,
borough: "$_id",
```

```
topRestaurants: {$slice: ["$topRestaurants", 5]}
  }}
])
```

87. Write a MongoDB query to find the borough with the highest number of restaurants that have a grade of "A" and a score greater than or equal to 90.

```
db.restaurants.aggregate([
  {
    $match: {
      "grades.grade": "A",
      "grades.score": { $gte: 90 }
    }
  },
  {
    $group: {
      _id: "$borough",
count: { $sum: 1 }
    }
  },
  {
    $sort: { count: -1 }
  },
  {
    $limit: 1
  }
]);
```