Tutorial – 1

1. Design html page to get below given output using bootstrap css.

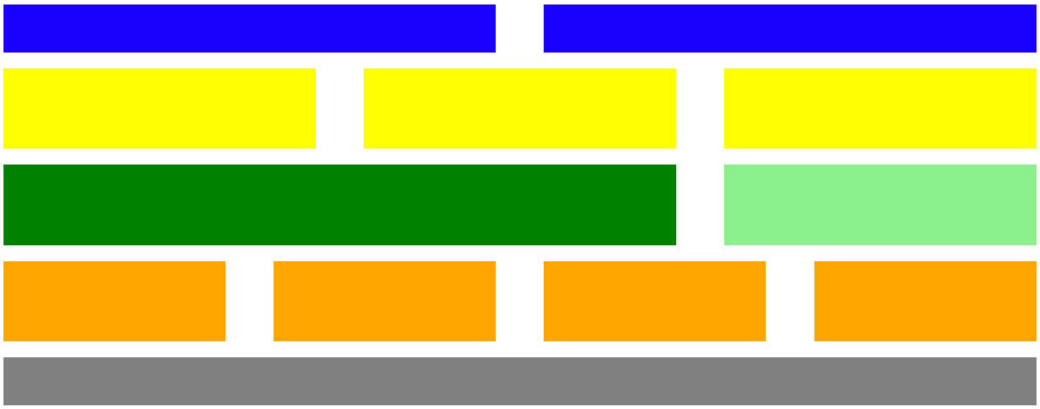
CODE :

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="utf-8">
  <title>UI</title>
  <link rel="stylesheet" href="../css/bootstrap.min.css">
  <script src="../js/bootstrap.bundle.min.js"></script>

  <style type="text/css">
  .primary{
   background: #1900ff; color: #1900ff;
  }
  .warning{
   background: #feff00; color: #feff00;
  }
  .success{
   background: #008100; color: #008100;
  }
  .cyan{
   background: #8CF08C; color: #8CF08C;
  }
  .oreng{
   background: #ffa700; color: #ffa700;
  }
  .gray{
   background: #808080; color: #808080;
  }
  </style>
 </head>
 <body>
     <div class="container">
     <div class="row mt-4">
            <div class="mr-2 col-6 col-sm-6 col-md-6 col-lg-6 col-xl-6">
                <div class="primary text-center">6</div>
            </div>
            <div class="ml-2 col-6 col-sm-6 col-md-6 col-lg-6 col-xl-6">
                <div class="primary text-center">6</div>
            </div>
```

```
        </div>
        <div class="row mt-2">
                <div class="mr-2 col-4 col-sm-4 col-md-4 col-lg-4 col-xl-4">
                        <div class="warning p-2 text-center">4</div>
                </div>
                <div class="mr-2 col-4 col-sm-4 col-md-4 col-lg-4 col-xl-4">
                        <div class="warning p-2 text-center">4</div>
                </div>
                <div class="mr-2 col-4 col-sm-4 col-md-4 col-lg-4 col-xl-4">
                        <div class="warning p-2 text-center">4</div>
                </div>
        </div>
        <div class="row mt-2">
                <div class="mr-2 col-8 col-sm-8 col-md-8 col-lg-8 col-xl-8">
                        <div class="success p-2 text-center">8</div>
                </div>
                <div class="mr-2 col-4 col-sm-4 col-md-4 col-lg-4 col-xl-4">
                        <div class="cyan p-2 text-center">4</div>
                </div>
        </div>
        <div class="row mt-2">
                <div class="mr-2 col-3 col-sm-3 col-md-3 col-lg-3 col-xl-3">
                        <div class="oreng p-2 text-center">3</div>
                </div>
                <div class="mr-2 col-3 col-sm-3 col-md-3 col-lg-3 col-xl-3">
                        <div class="oreng p-2 text-center">3</div>
                </div>
                <div class="mr-2 col-3 col-sm-3 col-md-3 col-lg-3 col-xl-3">
                        <div class="oreng p-2 text-center">3</div>
                </div>
                <div class="mr-2 col-3 col-sm-3 col-md-3 col-lg-3 col-xl-3">
                        <div class="oreng p-2 text-center">3</div>
                </div>
        </div>
        <div class="row mt-2">
                <div class="mr-2 col-12 col-sm-12 col-md-12 col-lg-12 col-xl-12">
                        <div class="gray text-center">12</div>
                </div>
        </div>
      </div>
    </body>
  </html>
```

::OUTPUT::



2. Use a Bootstrap class to style the table properly and get the following output (with padding and horizontal dividers).

    1. Add zebra-stripes to the table.
    2. Add borders on all sides of the table and cells.
    3. Enable a hover state on table rows.
    4. Make the table more compact by cutting cell padding in half.
    5. Use contextual classes to add the following:

    - Green color to the table row containing John.
    - Red color to the table row containing Mary.
    - Orange color to the last table row.

    CODE :

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta charset="utf-8">
  <title>UI</title>
  <link rel="stylesheet" href="../css/bootstrap.min.css">
  <script src="../js/bootstrap.bundle.min.js"></script>
 </head>
 <body>
 <div class="container">
 <table class="mt-4 table table-striped table-bordered table-hover table-sm">
   <thead>
    <tr>
     <th scope="col">Firstname</th>
     <th scope="col">Lastname</th>
```

```
      <th scope="col">Email</th>
     </tr>
    </thead>
    <tbody>
     <tr class="table-success">
      <td>John</td>
      <td>Doe</td>
      <td>john@example.com</td>
     </tr>
     <tr class="table-danger">
      <td>Mary</td>
      <td>Moe</td>
      <td>mary@example.cpm</td>
     </tr>
     <tr class="table-warning">
      <td>July</td>
      <td>Dooley</td>
      <td>july@example.com</td>
     </tr>
    </tbody>
   </table>
   </div>
  </body>
</html>
```

::OUTPUT::

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.cpm
July	Dooley	july@example.com

3. Bootstrapping with Buttons.

   Use a Bootstrap class to style the button properly with a red color.

   b. Change the size of the buttons in the following order: large, medium, small and xsmall.
   c. Make the button span the entire width of the parent element.
   d. Use a Bootstrap class to disable the button.

CODE :

```
<!DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="utf-8">
   <title>UI</title>
   <link rel="stylesheet" href="../css/bootstrap.min.css">
   <script src="../js/bootstrap.bundle.min.js"></script>
</head>
 <body>
       <div class="container">
<div class="mt-4 text-center">
    <button type="button" class="col-12 col-sm-4 col-md-4 col-lg-6 col-xl-6 btn btn-danger btn-
block" disabled>Danger</button>
   </div>
       </div>
 </body>
</html>
```

::OUTPUT::

4. Style the below given html form using bootstrap to get the output shown below.

CODE :

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="UTF-8">
 <title>Exercise #6: Simple form</title>
  <link rel="stylesheet" href="../css/bootstrap.min.css">
  <script src="../js/bootstrap.bundle.min.js"></script>

  <style type="text/css">
   .border{
    border-color: #E2E2E2;
    border-bottom-left-radius: 5px;
    border-bottom-right-radius: 5px;
    border-bottom-width: 5px;
    border-top-left-radius: 5px;
    border-top-right-radius: 5px;
   }
   .font-weight{
     font-weight: bold;
   }
   .margin-right{
    margin-right: 5px;
   }
  </style>
</head>
<body>
 <form class="form col-12 p-4" action="#" >
  <div class="mt-2">
   <label class="font-weight col-12" for="first_name">First name:</label>
   <input class="mt-1 col-12 border p-1" type="text" name="first_name" id="first_name"/>
  </div>
  <div class="mt-2">
   <label class="font-weight col-12" for="last_name">Last name:</label>
   <input class="mt-1 col-12 border p-1" type="text" name="last_name" id="last_name"/>
  </div>
  <div class="mt-2">
   <label><input class="margin-right" type="radio" name="gender" value="male"/>male</label>
   <label><input class="margin-right" type="radio" name="gender"
value="female"/>female</label>
  </div>
```

```
    <div class="mt-2">
      <label class="font-weight col-12" for="birth_date">Date of birth:</label>
      <input class="mt-1 col-12 border p-1" type="date" name="birth_date" id="birth_date"/>
    </div>
    <div class="mt-3">
      <input class="btn pl-2 pr-2 btn-primary text-center" type="submit" value="Add"/>
    </div>
  </form>
</body>
</html>
```

::OUTPUT::

Tutorial-2

```
C:\Users\bharg>mongoimport --db restaurant --collection res --file E:\restaurants.json
2022-01-28T16:10:40.236+0530    connected to: mongodb://localhost/
2022-01-28T16:10:40.792+0530    3772 document(s) imported successfully. 0 document(s) failed to import.
```

```
> show dbs
Students    0.000GB
admin       0.000GB
config      0.000GB
local       0.000GB
reg         0.000GB
restaurant  0.001GB
```

```
> use restaurant
switched to db restaurant
> show collections
res
```

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

CODE : AND : OUTPUT :

```
> db.res.find().pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305c88"),
        "address" : {
                "building" : "351",
                "coord" : [
                        -73.98513559999999,
                        40.7676919
                ],
                "street" : "West   57 Street",
                "zipcode" : "10019"
        },
        "borough" : "Manhattan",
        "cuisine" : "Irish",
        "grades" : [
                {
                        "date" : ISODate("2014-09-06T00:00:00Z"),
                        "grade" : "A",
                        "score" : 2
                },
                {
                        "date" : ISODate("2013-07-22T00:00:00Z"),
                        "grade" : "A",
                        "score" : 11
                },
                {
                        "date" : ISODate("2012-07-31T00:00:00Z"),
                        "grade" : "A",
                        "score" : 12
                },
                {
                        "date" : ISODate("2011-12-29T00:00:00Z"),
                        "grade" : "A",
                        "score" : 12
                }
        ],
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{"_id" : ...
```

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

CODE : AND : OUTPUT :

```
> db.res.find({},{restaurant_id:1,name:1,borough:1,cuisine:1}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305c88"),
        "borough" : "Manhattan",
        "cuisine" : "Irish",
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{
        "_id" : ObjectId("61f3c82886916f00ad305c89"),
        "borough" : "Brooklyn",
        "cuisine" : "Hamburgers",
        "name" : "Wendy'S",
        "restaurant_id" : "30112340"
}
{
        "_id" : ObjectId("61f3c82886916f00ad305c8a"),
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Riviera Caterer",
        "restaurant_id" : "40356018"
}
{
        "_id" : ObjectId("61f3c82886916f00ad305c8b"),
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "name" : "Morris Park Bake Shop",
        "restaurant_id" : "30075445"
}
{
        "_id" : ObjectId("61f3c82886916f00ad305c8c"),
        "borough" : "Staten Island",
        "cuisine" : "Jewish/Kosher",
        "name" : "Kosher Island",
        "restaurant_id" : "40356442"
}
{
```

7. 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.

CODE : AND : OUTPUT :

```
> db.res.find({},{restaurant_id:1,name:1,borough:1,cuisine:1,_id:0}).pretty()
{
        "borough" : "Manhattan",
        "cuisine" : "Irish",
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "Hamburgers",
        "name" : "Wendy'S",
        "restaurant_id" : "30112340"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Riviera Caterer",
        "restaurant_id" : "40356018"
}
{
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "name" : "Morris Park Bake Shop",
        "restaurant_id" : "30075445"
}
{
        "borough" : "Staten Island",
        "cuisine" : "Jewish/Kosher",
        "name" : "Kosher Island",
        "restaurant_id" : "40356442"
}
{
        "borough" : "Queens",
        "cuisine" : "Jewish/Kosher",
        "name" : "Tov Kosher Kitchen",
        "restaurant_id" : "40356068"
}
{
```

8. 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.

CODE : AND : OUTPUT :

```
> db.res.find({},{restaurant_id:1,name:1,borough:1,"address.zipcode":1,_id:0}).pretty()
{
        "address" : {
                "zipcode" : "10019"
        },
        "borough" : "Manhattan",
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{
        "address" : {
                "zipcode" : "11225"
        },
        "borough" : "Brooklyn",
        "name" : "Wendy'S",
        "restaurant_id" : "30112340"
}
{
        "address" : {
                "zipcode" : "11224"
        },
        "borough" : "Brooklyn",
        "name" : "Riviera Caterer",
        "restaurant_id" : "40356018"
}
{
        "address" : {
                "zipcode" : "10462"
        },
        "borough" : "Bronx",
        "name" : "Morris Park Bake Shop",
        "restaurant_id" : "30075445"
}
{
```

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

CODE : AND : OUTPUT :

```
> db.res.find({borough:"Bronx"}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305c8b"),
        "address" : {
                "building" : "1007",
                "coord" : [
                        -73.856077,
                        40.848447
                ],
                "street" : "Morris Park Ave",
                "zipcode" : "10462"
        },
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "grades" : [
                {
                        "date" : ISODate("2014-03-03T00:00:00Z"),
                        "grade" : "A",
                        "score" : 2
                },
                {
                        "date" : ISODate("2013-09-11T00:00:00Z"),
                        "grade" : "A",
                        "score" : 6
                },
                {
                        "date" : ISODate("2013-01-24T00:00:00Z"),
                        "grade" : "A",
                        "score" : 10
                },
                {
                        "date" : ISODate("2011-11-23T00:00:00Z"),
                        "grade" : "A",
                        "score" : 9
                },
                {
                        "date" : ISODate("2011-03-10T00:00:00Z"),
                        "grade" : "B",
                        "score" : 14
                }
        ],
```

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

CODE : AND : OUTPUT :

```
> db.res.find({borough:"Bronx"}).limit(5).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305c8b"),
        "address" : {
                "building" : "1007",
                "coord" : [
                        -73.856077,
                        40.848447
                ],
                "street" : "Morris Park Ave",
                "zipcode" : "10462"
        },
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "grades" : [
                {
                        "date" : ISODate("2014-03-03T00:00:00Z"),
                        "grade" : "A",
                        "score" : 2
                },
                {
                        "date" : ISODate("2013-09-11T00:00:00Z"),
                        "grade" : "A",
                        "score" : 6
                },
                {
                        "date" : ISODate("2013-01-24T00:00:00Z"),
                        "grade" : "A",
                        "score" : 10
                },
                {
                        "date" : ISODate("2011-11-23T00:00:00Z"),
                        "grade" : "A",
                        "score" : 9
                },
                {
                        "date" : ISODate("2011-03-10T00:00:00Z"),
                        "grade" : "B",
                        "score" : 14
                }
        ],
```

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

CODE : AND : OUTPUT :

```
> db.res.find({borough:"Bronx"}).limit(5).skip(5).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305cc4"),
        "address" : {
                "building" : "658",
                "coord" : [
                        -73.81363999999999,
                        40.82941100000001
                ],
                "street" : "Clarence Ave",
                "zipcode" : "10465"
        },
        "borough" : "Bronx",
        "cuisine" : "American ",
        "grades" : [
                {
                        "date" : ISODate("2014-06-21T00:00:00Z"),
                        "grade" : "A",
                        "score" : 5
                },
                {
                        "date" : ISODate("2012-07-11T00:00:00Z"),
                        "grade" : "A",
                        "score" : 10
                }
        ],
        "name" : "Manhem Club",
        "restaurant_id" : "40364363"
}
{
        "_id" : ObjectId("61f3c82886916f00ad305cde"),
        "address" : {
                "building" : "2222",
                "coord" : [
                        -73.84971759999999,
                        40.8304811
                ],
                "street" : "Haviland Avenue",
                "zipcode" : "10462"
        },
```

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

CODE : AND : OUTPUT :

```
> db.res.find({grades:{$elemMatch:{score:{$gt:90}}}},{"grades.$":1}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305de5"),
        "grades" : [
                {
                        "date" : ISODate("2014-03-28T00:00:00Z"),
                        "grade" : "C",
                        "score" : 131
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad305e87"),
        "grades" : [
                {
                        "date" : ISODate("2012-04-06T00:00:00Z"),
                        "grade" : "C",
                        "score" : 92
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad305fe9"),
        "grades" : [
                {
                        "date" : ISODate("2014-06-17T00:00:00Z"),
                        "grade" : "C",
                        "score" : 98
                }
        ]
}
>
```

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

CODE : AND : OUTPUT :

```
> db.res.find({grades:{$elemMatch:{score:{$gt:80,$lt:100}}}},{"grades.$":1}).pretty
()
{
        "_id" : ObjectId("61f3c82886916f00ad305e87"),
        "grades" : [
                {
                        "date" : ISODate("2012-04-06T00:00:00Z"),
                        "grade" : "C",
                        "score" : 92
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad305fe9"),
        "grades" : [
                {
                        "date" : ISODate("2014-06-17T00:00:00Z"),
                        "grade" : "C",
                        "score" : 98
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad306855"),
        "grades" : [
                {
                        "date" : ISODate("2014-06-27T00:00:00Z"),
                        "grade" : "C",
                        "score" : 89
                }
        ]
}
>
```

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

CODE : AND : OUTPUT :

```
> db.res.find({"address.coord.1":{$lt:-25.754168}}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305d70"),
        "address" : {
                "building" : "155157",
                "coord" : [
                        153.1628795,
                        -28.0168595
                ],
                "street" : "Christie St.",
                "zipcode" : "10002"
        },
        "borough" : "Manhattan",
        "cuisine" : "Steak",
        "grades" : [
                {
                        "date" : ISODate("2014-11-12T00:00:00Z"),
                        "grade" : "A",
                        "score" : 12
                },
                {
                        "date" : ISODate("2013-09-24T00:00:00Z"),
                        "grade" : "A",
                        "score" : 7
                },
                {
                        "date" : ISODate("2013-04-12T00:00:00Z"),
                        "grade" : "B",
                        "score" : 26
                },
                {
                        "date" : ISODate("2012-09-21T00:00:00Z"),
                        "grade" : "A",
                        "score" : 5
                },
                {
                        "date" : ISODate("2012-04-10T00:00:00Z"),
                        "grade" : "A",
                        "score" : 2
                }
        ],
        "name" : "Sammy'S Steakhouse",
        "restaurant_id" : "40368552"
}
>
```

15. 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.

CODE : AND : OUTPUT :

```
> db.res.find({$and:[{cuisine:{$ne:"American"}},{grades:{$elemMatch:{score:{$gt:70}}}},{"address.coord.1":{$lt:40.7308729}}]},{cuisine:1,"grades.$":1,"address.coord":1}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305e87"),
        "address" : {
                "coord" : [
                        -73.9864626,
                        40.7266739
                ]
        },
        "cuisine" : "Indian",
        "grades" : [
                {
                        "date" : ISODate("2012-04-06T00:00:00Z"),
                        "grade" : "C",
                        "score" : 92
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad306145"),
        "address" : {
                "coord" : [
                        -73.94610279999999,
                        40.7137587
                ]
        },
        "cuisine" : "Bakery",
        "grades" : [
                {
                        "date" : ISODate("2011-05-03T00:00:00Z"),
                        "grade" : "C",
                        "score" : 77
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad306855"),
        "address" : {
                "coord" : [
                        -74.0163793,
                        40.7167671
```

16. 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.

CODE : AND : OUTPUT :

```
> db.res.find({cuisine:{$ne:"American"},grades:{$elemMatch:{score:{$gt:70}}},"add
ress.coord.0":{$lt:-65.754168}},{cuisine:1,"grades.$":1,"address.coord":1}).prett
y()
{
        "_id" : ObjectId("61f3c82886916f00ad305de5"),
        "address" : {
                "coord" : [
                        -73.9782725,
                        40.7624022
                ]
        },
        "cuisine" : "American ",
        "grades" : [
                {
                        "date" : ISODate("2014-03-28T00:00:00Z"),
                        "grade" : "C",
                        "score" : 131
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad305e87"),
        "address" : {
                "coord" : [
                        -73.9864626,
                        40.7266739
                ]
        },
        "cuisine" : "Indian",
        "grades" : [
                {
                        "date" : ISODate("2012-04-06T00:00:00Z"),
                        "grade" : "C",
                        "score" : 92
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad305f15"),
        "address" : {
                "coord" : [
                        -73.9883909,
                        40.740735
                ]
        },
        "cuisine" : "American ",
```

17. 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.

CODE : AND : OUTPUT :

```
> db.res.find({cuisine:{$ne:"American"},grades:{$elemMatch:{grade:"A"}},borough:{
$ne:"Brooklyn"}},{cuisine:1,"grades.$":1,borough:1}).sort({cuisine:-1}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad306393"),
        "borough" : "Manhattan",
        "cuisine" : "Vietnamese/Cambodian/Malaysia",
        "grades" : [
                {
                        "date" : ISODate("2014-08-21T00:00:00Z"),
                        "grade" : "A",
                        "score" : 13
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad30644d"),
        "borough" : "Queens",
        "cuisine" : "Vietnamese/Cambodian/Malaysia",
        "grades" : [
                {
                        "date" : ISODate("2013-05-20T00:00:00Z"),
                        "grade" : "A",
                        "score" : 13
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad306836"),
        "borough" : "Manhattan",
        "cuisine" : "Vietnamese/Cambodian/Malaysia",
        "grades" : [
                {
                        "date" : ISODate("2014-10-01T00:00:00Z"),
                        "grade" : "A",
                        "score" : 7
                }
        ]
}
{
        "_id" : ObjectId("61f3c82886916f00ad305f37"),
        "borough" : "Manhattan",
        "cuisine" : "Vegetarian",
        "grades" : [
                {
                        "date" : ISODate("2014-02-05T00:00:00Z"),
                        "grade" : "A",
```

18. 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.

CODE : AND : OUTPUT :

```
> db.res.find({name:{$regex:/^Wil/}},{restaurant_id:1,name:1,cuisine:1,borough:1,
_id:0}).pretty()
{
        "borough" : "Bronx",
        "cuisine" : "American ",
        "name" : "Wild Asia",
        "restaurant_id" : "40357217"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "Delicatessen",
        "name" : "Wilken'S Fine Food",
        "restaurant_id" : "40356483"
}
{
        "borough" : "Bronx",
        "cuisine" : "Pizza",
        "name" : "Wilbel Pizza",
        "restaurant_id" : "40871979"
}
```

19. 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.

CODE : AND : OUTPUT :

```
> db.res.find({name:{$regex:/ces$/}},{restaurant_id:1,name:1,cuisine:1,borough:1,
_id:0}).pretty()
{
        "borough" : "Manhattan",
        "cuisine" : "American ",
        "name" : "Pieces",
        "restaurant_id" : "40399910"
}
{
        "borough" : "Queens",
        "cuisine" : "American ",
        "name" : "S.M.R Restaurant Services",
        "restaurant_id" : "40403857"
}
{
        "borough" : "Manhattan",
        "cuisine" : "American ",
        "name" : "Good Shepherd Services",
        "restaurant_id" : "40403989"
}
{
        "borough" : "Queens",
        "cuisine" : "Ice Cream, Gelato, Yogurt, Ices",
        "name" : "The Ice Box-Ralph'S Famous Italian Ices",
        "restaurant_id" : "40690899"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "Jewish/Kosher",
        "name" : "Alices",
        "restaurant_id" : "40782042"
}
{
        "borough" : "Manhattan",
        "cuisine" : "American ",
        "name" : "Re: Sources",
        "restaurant_id" : "40876068"
}
```

20. 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.

CODE : AND : OUTPUT :

```
> db.res.find({name:{$regex:/Reg/}},{restaurant_id:1,name:1,cuisine:1,borough:1,_
id:0}).pretty()
{
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Regina Caterers",
        "restaurant_id" : "40356649"
}
{
        "borough" : "Manhattan",
        "cuisine" : "Café/Coffee/Tea",
        "name" : "Caffe Reggio",
        "restaurant_id" : "40369418"
}
{
        "borough" : "Manhattan",
        "cuisine" : "American ",
        "name" : "Regency Hotel",
        "restaurant_id" : "40382679"
}
{
        "borough" : "Manhattan",
        "cuisine" : "American ",
        "name" : "Regency Whist Club",
        "restaurant_id" : "40402377"
}
{
        "borough" : "Queens",
        "cuisine" : "American ",
        "name" : "Rego Park Cafe",
        "restaurant_id" : "40523342"
}
{
        "borough" : "Queens",
        "cuisine" : "Pizza",
        "name" : "Regina Pizza",
        "restaurant_id" : "40801325"
}
{
        "borough" : "Manhattan",
        "cuisine" : "American ",
        "name" : "Regal Entertainment Group",
        "restaurant_id" : "40891782"
}
```

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

CODE : AND : OUTPUT :

```
> db.res.find({borough:"Bronx",cuisine:{$in:["American","Chinese"]}},{restaurant_
id:1,name:1,cuisine:1,borough:1,_id:0}).pretty()
{
        "borough" : "Bronx",
        "cuisine" : "Chinese",
        "name" : "Happy Garden",
        "restaurant_id" : "40363289"
}
{
        "borough" : "Bronx",
        "cuisine" : "Chinese",
        "name" : "Happy Garden",
        "restaurant_id" : "40364296"
}
{
        "borough" : "Bronx",
        "cuisine" : "Chinese",
        "name" : "China Wok Ii",
        "restaurant_id" : "40510328"
}
{
        "borough" : "Bronx",
        "cuisine" : "Chinese",
        "name" : "Dragon City",
        "restaurant_id" : "40529203"
}
{
        "borough" : "Bronx",
        "cuisine" : "Chinese",
        "name" : "Hunan Balcony",
        "restaurant_id" : "40551996"
}
{
        "borough" : "Bronx",
        "cuisine" : "Chinese",
        "name" : "Great Wall Restaurant",
        "restaurant_id" : "40552226"
}
{
        "borough" : "Bronx",
        "cuisine" : "Chinese",
        "name" : "Lucky House Restaurant",
        "restaurant_id" : "40571587"
}
{
```

22. 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 Bronxor Brooklyn.

CODE : AND : OUTPUT :

```
> db.res.find({borough:{$in:["Ststen Island","Queens","Bronxor Brooklyn"]}},{rest
aurant_id:1,name:1,cuisine:1,borough:1,_id:0}).pretty()
{
        "borough" : "Queens",
        "cuisine" : "Jewish/Kosher",
        "name" : "Tov Kosher Kitchen",
        "restaurant_id" : "40356068"
}
{
        "borough" : "Queens",
        "cuisine" : "American ",
        "name" : "Brunos On The Boulevard",
        "restaurant_id" : "40356151"
}
{
        "borough" : "Queens",
        "cuisine" : "Ice Cream, Gelato, Yogurt, Ices",
        "name" : "Carvel Ice Cream",
        "restaurant_id" : "40361322"
}
{
        "borough" : "Queens",
        "cuisine" : "Delicatessen",
        "name" : "Sal'S Deli",
        "restaurant_id" : "40361618"
}
{
        "borough" : "Queens",
        "cuisine" : "Delicatessen",
        "name" : "Steve Chu'S Deli & Grocery",
        "restaurant_id" : "40361998"
}
{
        "borough" : "Queens",
        "cuisine" : "Chinese",
        "name" : "Ho Mei Restaurant",
        "restaurant_id" : "40362432"
}
{
        "borough" : "Queens",
        "cuisine" : "Delicatessen",
        "name" : "Tony'S Deli",
        "restaurant_id" : "40363333"
}
{
```

23. 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.

CODE : AND : OUTPUT :

```
> db.res.find({borough:{$nin:["Ststen Island","Queens","Bronxor Brooklyn"]}},{res
taurant_id:1,name:1,cuisine:1,borough:1,_id:0}).pretty()
{
        "borough" : "Manhattan",
        "cuisine" : "Irish",
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "Hamburgers",
        "name" : "Wendy'S",
        "restaurant_id" : "30112340"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Riviera Caterer",
        "restaurant_id" : "40356018"
}
{
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "name" : "Morris Park Bake Shop",
        "restaurant_id" : "30075445"
}
{
        "borough" : "Staten Island",
        "cuisine" : "Jewish/Kosher",
        "name" : "Kosher Island",
        "restaurant_id" : "40356442"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Regina Caterers",
        "restaurant_id" : "40356649"
}
{
        "borough" : "Bronx",
        "cuisine" : "American ",
        "name" : "Wild Asia",
        "restaurant_id" : "40357217"
}
```

24. 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'.

CODE : AND : OUTPUT :

```
> db.res.find({$or:[{$and:[{cuisine:{$ne:"American"}},{cuisine:{$ne:"Chinese"}}]}
,{name:{$regex:/^Wil/}}]},{restaurant_id:1,name:1,cuisine:1,borough:1,_id:0}).pre
tty()
{
        "borough" : "Manhattan",
        "cuisine" : "Irish",
        "name" : "Dj Reynolds Pub And Restaurant",
        "restaurant_id" : "30191841"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "Hamburgers",
        "name" : "Wendy'S",
        "restaurant_id" : "30112340"
}
{
        "borough" : "Brooklyn",
        "cuisine" : "American ",
        "name" : "Riviera Caterer",
        "restaurant_id" : "40356018"
}
{
        "borough" : "Bronx",
        "cuisine" : "Bakery",
        "name" : "Morris Park Bake Shop",
        "restaurant_id" : "30075445"
}
{
        "borough" : "Staten Island",
        "cuisine" : "Jewish/Kosher",
        "name" : "Kosher Island",
        "restaurant_id" : "40356442"
}
{
        "borough" : "Queens",
        "cuisine" : "Jewish/Kosher",
        "name" : "Tov Kosher Kitchen",
        "restaurant_id" : "40356068"
}
{
        "borough" : "Queens",
        "cuisine" : "American ",
        "name" : "Brunos On The Boulevard",
```

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

CODE : AND : OUTPUT :

```
> db.res.find().sort({name:1}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad306917"),
        "address" : {
                "building" : "129",
                "coord" : [
                        -73.962943,
                        40.685007
                ],
                "street" : "Gates Avenue",
                "zipcode" : "11238"
        },
        "borough" : "Brooklyn",
        "cuisine" : "Italian",
        "grades" : [
                {
                        "date" : ISODate("2014-03-06T00:00:00Z"),
                        "grade" : "A",
                        "score" : 5
                },
                {
                        "date" : ISODate("2013-08-29T00:00:00Z"),
                        "grade" : "A",
                        "score" : 2
                },
                {
                        "date" : ISODate("2013-03-08T00:00:00Z"),
                        "grade" : "A",
                        "score" : 7
                },
                {
                        "date" : ISODate("2012-06-27T00:00:00Z"),
                        "grade" : "A",
                        "score" : 7
                },
                {
                        "date" : ISODate("2011-11-17T00:00:00Z"),
                        "grade" : "A",
                        "score" : 12
                }
        ],
        "name" : "(Lewis Drug Store) Locanda Vini E Olii",
        "restaurant_id" : "40804423"
}
```

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

CODE : AND : OUTPUT :

```
> db.res.find().sort({name:-1}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305d47"),
        "address" : {
                "building" : "6946",
                "coord" : [
                        -73.8811834,
                        40.7017759
                ],
                "street" : "Myrtle Avenue",
                "zipcode" : "11385"
        },
        "borough" : "Queens",
        "cuisine" : "German",
        "grades" : [
                {
                        "date" : ISODate("2014-09-24T00:00:00Z"),
                        "grade" : "A",
                        "score" : 11
                },
                {
                        "date" : ISODate("2014-04-17T00:00:00Z"),
                        "grade" : "A",
                        "score" : 7
                },
                {
                        "date" : ISODate("2013-03-12T00:00:00Z"),
                        "grade" : "A",
                        "score" : 13
                },
                {
                        "date" : ISODate("2012-10-02T00:00:00Z"),
                        "grade" : "A",
                        "score" : 9
                },
                {
                        "date" : ISODate("2012-05-09T00:00:00Z"),
                        "grade" : "A",
                        "score" : 13
                },
                {
                        "date" : ISODate("2011-12-28T00:00:00Z"),
                        "grade" : "B",
                        "score" : 24
```

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

CODE : AND : OUTPUT :

```
> db.res.find().sort({suisine:1,borough:-1}).pretty()
{
        "_id" : ObjectId("61f3c82886916f00ad305c8c"),
        "address" : {
                "building" : "2206",
                "coord" : [
                        -74.1377286,
                        40.6119572
                ],
                "street" : "Victory Boulevard",
                "zipcode" : "10314"
        },
        "borough" : "Staten Island",
        "cuisine" : "Jewish/Kosher",
        "grades" : [
                {
                        "date" : ISODate("2014-10-06T00:00:00Z"),
                        "grade" : "A",
                        "score" : 9
                },
                {
                        "date" : ISODate("2014-05-20T00:00:00Z"),
                        "grade" : "A",
                        "score" : 12
                },
                {
                        "date" : ISODate("2013-04-04T00:00:00Z"),
                        "grade" : "A",
                        "score" : 12
                },
                {
                        "date" : ISODate("2012-01-24T00:00:00Z"),
                        "grade" : "A",
                        "score" : 9
                }
        ],
        "name" : "Kosher Island",
        "restaurant_id" : "40356442"
}
```