

# Задание 1:

## Задание 1

Выполните импорт коллекции из файла `restaurants.json`

### Структура коллекции ресторанов

```
{
  "address": {
    «building»: «1007»,
    «coord»: [-73,856077, 40,848447],
    "street": "Моррис Парк Авеню",
    "zipcode": 10462
  },
  "borough": "Бронкс",
  "cuisine": "пекарня",
  "grades": [
    {"date": {"$ date": 1393804800000}, "grade": "A", "score": 2},
    {"date": {"$ date": 1378857600000}, "grade": "A", "score": 6},
    {"date": {"$ date": 1358985600000}, "grade": "A", "score": 10},
    {"date": {"$ date": 1322006400000}, "grade": "A", "score": 9},
    {"date": {"$ date": 1299715200000}, "grade": "B", "score": 14}
  ],
  "name": "Morris Park Bake Shop",
  "restaurant_id": "30075445"
}
```

1 Выведите все документы коллекции Ресторан в формате: **restaurant\_id**, **name**, **borough** и **cuisine**, вывод **\_id** для всех документов исключить.

```
> db.restaurants.find(
  {},
  {
    _id: 0,
    restaurant_id: 1,
    name: 1,
    borough: 1,
    cuisine: 1
  }
).limit(5)
< {
  borough: 'Bronx',
  cuisine: 'Bakery',
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
{
  borough: 'Brooklyn',
  cuisine: 'Hamburgers',
  name: 'Wendy'S',
  restaurant_id: '30112340'
}
```

2 Выведите первые 5 ресторанов в алфавитном порядке, которые находятся в районе **Bronx**.

```
> db.restaurants.find(
  { borough: "Bronx" },
  {
    _id: 0,
    restaurant_id: 1,
    name: 1,
    borough: 1,
    cuisine: 1
  }
)
.sort({ name: 1 })
.limit(5)
< {
  borough: 'Bronx',
  cuisine: 'American ',
  name: 'African Market (Baboon Cafe)',
  restaurant_id: '40368026'
}
{
  borough: 'Bronx',
  cuisine: 'African',
  name: 'African Terrace',
  restaurant_id: '40368021'
}
```

3 Найдите рестораны, которые набрали более 80, но менее 100 баллов.

```
> db.restaurants.find({
  "grades": {
    $elemMatch: {
      "score": { $gt: 80, $lt: 100 }
    }
  }
},
{
  _id: 0,
  restaurant_id: 1,
  name: 1,
  borough: 1,
  cuisine: 1,
  "grades.score": 1
}).limit(10)
< {
  borough: 'Manhattan',
  cuisine: 'Indian',
  grades: [
    {

```

```
{  
  borough: 'Manhattan',  
  cuisine: 'Indian',  
  grades: [  
    {  
      score: 5  
    },  
    {  
      score: 8  
    },  
    {  
      score: 12  
    },  
    {  
      score: 2  
    },  
    {  
      score: 9  
    },  
    {  
      score: 92  
    },  
    {  
      score: 41  
    }  
  ],  
  name: 'Gandhi',  
  restaurant_id: '40381295'
```

4 Найдите рестораны, которые не относятся к типу кухни **American**, получили оценку «**A**», не расположены в районе **Brooklyn**. Документ должен отображаться в соответствии с кухней в порядке убывания.

```
> db.restaurants.find({
  cuisine: { $ne: "American" },
  "grades.grade": "A",
  borough: { $ne: "Brooklyn" }
},
{
  _id: 0,
  restaurant_id: 1,
  name: 1,
  borough: 1,
  cuisine: 1
})
.sort({ cuisine: -1 })
.limit(10)
< {
```

```
< {  
  borough: 'Manhattan',  
  cuisine: 'Vietnamese/Cambodian/Malaysia',  
  name: 'Nha-Trang Centre Vietnam Restaurant',  
  restaurant_id: '40751226'  
}  
{  
  borough: 'Manhattan',  
  cuisine: 'Vietnamese/Cambodian/Malaysia',  
  name: 'Thai Son',  
  restaurant_id: '40559606'  
}  
{  
  borough: 'Queens',  
  cuisine: 'Vietnamese/Cambodian/Malaysia',  
  name: 'Pho Bac Vietnamese Seafood Cuisine',  
  restaurant_id: '40578058'  
}  
{  
  borough: 'Manhattan',  
  cuisine: 'Vegetarian',  
  name: 'Candle Cafe',
```

5 Найдите идентификатор ресторана, название, район и кухню для тех ресторанов, чье название начинается с первых трех букв назвали «**Wil**»

```
> db.restaurants.find(  
  { name: /^Wil/ },  
  {  
    _id: 0,  
    restaurant_id: 1,  
    name: 1,  
    borough: 1,  
    cuisine: 1  
  }  
)  
< {
```



```
< {  
  borough: 'Brooklyn',  
  cuisine: 'Delicatessen',  
  name: "Wilken'S Fine Food",  
  restaurant_id: '40356483'  
}  
{  
  borough: 'Bronx',  
  cuisine: 'American ',  
  name: 'Wild Asia',  
  restaurant_id: '40357217'  
}  
{  
  borough: 'Bronx',  
  cuisine: 'Pizza',  
  name: 'Wilbel Pizza',  
  restaurant_id: '40871979'  
}  
test> |
```

6 Найдите рестораны, которые относятся к району **Bronx** и готовят **American** или **Chinese** блюда.

```
> db.restaurants.find({
  borough: "Bronx",
  cuisine: { $in: ["American", "Chinese"] }
},
{
  _id: 0,
  restaurant_id: 1,
  name: 1,
  borough: 1,
  cuisine: 1
}).limit(10)
```

```
}).limit(10)
< {
  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'
}
```

7 Найдите идентификатор ресторана, название и оценки для тех ресторанов, которые «**2014-08-11T00: 00: 00Z**» набрали **9** баллов за оценку **A**

```
> db.restaurants.find({
  "grades": {
    $elemMatch: {
      "grade": "A",
      "score": 9,
      "date.$date": 1407715200000
    }
  }
}, {
  _id: 0,
  restaurant_id: 1,
  name: 1,
  grades: 1
})
```

```
< {
  grades: [
    {
      date: 2014-08-11T00:00:00.000Z,
      grade: 'A',
      score: 9
    },
  ],
}
```

8 В каждом районе посчитайте количество ресторанов по каждому виду кухни. Документ должен иметь формат ***borough, cuisine, count***

```
> db.restaurants.aggregate([
  {
    $group:{
      _id: {borough: "$borough", cuisine: "$cuisine"},
      count: { $sum: 1 }
    }
  }
])
< {
  _id: {
    borough: 'Brooklyn',
    cuisine: 'Pizza/Italian'
  },
  count: 42
}
{
  _id: {
    borough: 'Manhattan',
    cuisine: 'Other'
  },
  count: 3
}
```

9 В районе **Bronx** найдите ресторан с минимальной суммой набранных баллов.

```
> db.restaurants.aggregate([
  { $match: { borough: "Bronx" } },
  {
    $addFields: {
      totalScore: {
        $sum: "$grades.score"
      }
    }
  },
  { $sort: { totalScore: 1 } },
  { $limit: 1 },
  {
    $project: {
      _id: 0,
      restaurant_id: 1,
      name: 1,
      borough: 1,
      totalScore: 1,
      "grades.score": 1
    }
  }
])
```

```
    ])  
    < {  
      borough: 'Bronx',  
      grades: [  
        {  
          score: 4  
        }  
      ],  
      name: 'Ambassador Diner',  
      restaurant_id: '40403946',  
      totalScore: 4  
    }
```

10 Добавьте в коллекцию свой любимый ресторан.

```
> db.restaurants.insertOne({
  "address": {
    "building": "15",
    "coord": [37.6173, 55.7558],
    "street": "Тверская улица",
    "zipcode": 125009
  },
  "borough": "Центральный",
  "cuisine": "Русская",
  "grades": [
    {
      "date": new Date("2024-01-15T00:00:00Z"),
      "grade": "A",
      "score": 10
    }
  ],
  "name": "Старик Хинкалыч",
  "restaurant_id": "325325325"
})
```

```
< {
  acknowledged: true,
  insertedId: ObjectId('6945b9522feb919ec90a7a5e')
}
```



11 В добавленном ресторане укажите информацию о времени его работы.

```
> db.restaurants.updateOne(
  { restaurant_id: "325325325" },
  {
    $set: {
      "opening_hours": {
        "monday": "10:00-23:00",
        "tuesday": "10:00-23:00",
        "wednesday": "10:00-23:00",
        "thursday": "10:00-23:00",
        "friday": "10:00-00:00",
        "saturday": "11:00-00:00",
        "sunday": "11:00-22:00"
      }
    }
  }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
test>
```

12 Измените время работы вашего любимого ресторана.

```
> db.restaurants.updateOne(
  { restaurant_id: "325325325" },
  {
    $set: {
      "opening_hours": {
        "monday": "09:00-22:00",
        "tuesday": "09:00-22:00",
        "wednesday": "09:00-22:00",
        "thursday": "09:00-22:00",
        "friday": "09:00-23:00",
        "saturday": "10:00-23:00",
        "sunday": "10:00-21:00"
      }
    }
  }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Проверка:

```
    }  
    > db.restaurants.findOne({ restaurant_id: "325325325" })  
    < {  
      _id: ObjectId('6945b9522feb919ec90a7a5e'),  
      address: {  
        building: '15',  
        coord: [  
          37.6173,  
          55.7558  
        ],  
        street: 'Тверская улица',  
        zipcode: 125009  
      },  
      borough: 'Центральный',  
      cuisine: 'Русская',  
      grades: [  
        {  
          date: 2024-01-15T00:00:00.000Z,  
          grade: 'A',  
          score: 10  
        }  
      ]  
    }
```

```
],  
  name: 'Старик Хинкалыч',  
  restaurant_id: '325325325',  
  opening_hours: {  
    monday: '09:00-22:00',  
    tuesday: '09:00-22:00',  
    wednesday: '09:00-22:00',  
    thursday: '09:00-22:00',  
    friday: '09:00-23:00',  
    saturday: '10:00-23:00',  
    sunday: '10:00-21:00'  
  }  
}
```

## Задание 2:

### Задание 2

Выполните импорт коллекции из файла `weather.json`

Структура коллекции Погода

```
{
  "year": 2014,
  "month": 1,
  "day": 1,
  "hour": 0,
  "temperature": -1,
  "wind_direction": "Южный",
  "wind": "3",
  "code": "CL",
  "clouds": 9,
  "visibility": 10,
  "humidity": 94,
  "pressure": 1030
}
```

```
{
  _id: ObjectId('6945c66c2feb919ec90a7a61'),
  year: 2014,
  month: 1,
  day: 1,
  hour: 6,
  temperature: -1.3,
  wind_direction: 'Южный',
  wind: '3',
  code: 'CL',
  clouds: 9,
  visibility: 10,
  humidity: 92,
  pressure: 1028
}
```

Активировать  
Чтобы активировать

1. Какова разница между максимальной и минимальной температурой в течение года?

```
> db.weather.aggregate([
  {
    $group: {
      _id: null,
      maxTemp: { $max: "$temperature" },
      minTemp: { $min: "$temperature" }
    }
  },
  {
    $project: {
      _id: 0,
      maxTemperature: "$maxTemp",
      minTemperature: "$minTemp",
      temperatureDifference: { $subtract: ["$maxTemp", "$minTemp"] }
    }
  }
])
< {
  maxTemperature: 34.8,
  minTemperature: -22.6,
  temperatureDifference: 57.4
}
test> |
```

Активация Windows

Чтобы активировать Windows, перейдите в  
"Параметры".

2. Какова средняя температура в году, если исключить 10 дней с самой низкой температурой и 10 дней с самой высокой?

```
> db.weather.aggregate([
  {
    $group: {
      _id: { year: "$year", month: "$month", day: "$day" },
      dailyAvgTemp: { $avg: "$temperature" }
    }
  },
  { $sort: { dailyAvgTemp: 1 } },
  {
    $group: {
      _id: null,
      allDays: { $push: "$dailyAvgTemp" },
      totalDays: { $sum: 1 }
    }
  },
  {
    $project: {
      filteredTemps: {
        $slice: ["$allDays", 10, { $subtract: ["$totalDays", 20] }]
      }
    }
  }
])
```

```

    $project: {
      filteredTemps: {
        $slice: ["$allDays", 10, { $subtract: ["$totalDays", 20] }]
      }
    },
    { $unwind: "$filteredTemps" },
    {
      $group: {
        _id: null,
        averageTemperature: { $avg: "$filteredTemps" }
      }
    },
    {
      $project: {
        _id: 0,
        averageTemperatureExcludingExtremes: { $round: ["$averageTemperature",
      }
    }
  })
}
< {
  averageTemperatureExcludingExtremes: 7.97
}

```



3. Найти первые 10 записей с самой низкой погодой, когда дул ветер с юга и посчитайте среднюю температуры для этих записей

```
> db.weather.aggregate([
  {
    $match: { wind_direction: "Южный" }
  },
  {
    $sort: { temperature: 1 }
  },
  {
    $limit: 10
  },
  {
    $group: {
      _id: null,
      averageTemperature: { $avg: "$temperature" },
      temperatures: { $push: "$temperature" }
    }
  },
  {
    $project: {
      _id: 0,
      averageTemperature: { $round: ["$averageTemperature", 2] }
    }
  }
])
< {
  averageTemperature: -12.95
```

Активат  
Чтобы акт

4. Подсчитайте количество дней, когда шел снег. (Будем считать снегом осадки, которые выпали, когда температура была ниже нуля)

```
db.weather.aggregate([
  { $match: { temperature: { $lt: 0 }, code: "SN" } },
  {
    $group: {
      _id: { year: "$year", month: "$month", day: "$day" }
    }
  },
  { $count: "snowyDays" }
])
{
  snowyDays: 26
}
```

5. В течение зимы иногда шел снег, а иногда дождь. Насколько больше (или меньше) выпало осадков в виде снега.

```
db.weather.aggregate([
  {
    $match: {
      month: { $in: [12, 1, 2] },
      code: { $in: ["SN", "RA"] }
    }
  },
  {
    $group: {
      _id: null,
      snowDays: {
        $sum: { $cond: [{ $eq: ["$code", "SN"] }, 1, 0] }
      },
      rainDays: {
        $sum: { $cond: [{ $eq: ["$code", "RA"] }, 1, 0] }
      }
    }
  },
])
```

```
{
  $project: {
    _id: 0,
    snowDays: 1,
    rainDays: 1,
    difference: { $subtract: ["$snowDays", "$rainDays"] }
  }
}
])
< {
  snowDays: 67,
  rainDays: 57,
  difference: 10
}
```

Активация Windows

Чтобы активировать Windows,  
перейдите в "Параметры".

6. Какова вероятность того что в ясный день выпадут осадки? (Предположим, что день считается ясным, если ясная погода фиксируется более чем в 75% случаев)

```
db.weather.aggregate([

  {
    $group: {
      _id: {
        year: "$year",
        month: "$month",
        day: "$day"
      },
      total: { $sum: 1 },
      clear: {
        $sum: { $cond: [{ $lt: ["$clouds", 7] }, 1, 0] }
      },
      hasPrecip: {
        $max: {
          $cond: [
            { $in: ["$code", ["RA", "SN", "SHRA", "TS", "BR"]] },
            1,
            0
          ]
        }
      }
    }
  }
],
```

```
{
  $match: {
    $expr: { $gt: [{ $divide: ["$clear", "$total"] }, 0.75] }
  },

  {
    $group: {
      _id: null,
      clearDays: { $sum: 1 },
      precipDays: { $sum: "$hasPrecip" }
    }
  },
}
```

```

{
  $project: {
    _id: 0,
    clearDays: 1,
    precipDays: 1,
    probability: {
      $round: [
        { $multiply: [
          { $divide: ["$precipDays", "$clearDays"] },
          100
        ]},
        2
      ]
    }
  }
}
])
< {
  clearDays: 68,
  precipDays: 12,
  probability: 17.65
}

```



Ножницы

Снимок экрана

Автоматически

Активация Windows

Чтобы активировать

7. Увеличьте температуру на один градус при каждом измерении в нечетный день во время зимы. На сколько градусов изменилась средняя температура?

```
//7
const beforeUpdate = db.weather.aggregate([
  {
    $match: {
      month: { $in: [12, 1, 2] }
    }
  },
  {
    $group: {
      _id: null,
      avgTempBefore: { $avg: "$temperature" }
    }
  }
]).toArray()[0];
```

```
db.weather.updateMany(
  {
    month: { $in: [12, 1, 2] },
    day: { $mod: [2, 1] }
  },
  {
    $inc: { temperature: 1 }
  }
);
```



```
const afterUpdate = db.weather.aggregate([
  {
    $match: {
      month: { $in: [12, 1, 2] }
    }
  },
  {
    $group: {
      _id: null,
      avgTempAfter: { $avg: "$temperature" }
    }
  }
]).toArray()[0];
```

```
const result = {
  averageTemperatureBefore: beforeUpdate.avgTempBefore.toFixed(2),
  averageTemperatureAfter: afterUpdate.avgTempAfter.toFixed(2),
  temperatureChange: (afterUpdate.avgTempAfter - beforeUpdate.avgTempBefore).toFixed(2)
};

printjson(result);

db.weather.updateMany(
  {
    month: { $in: [12, 1, 2] },
    day: { $mod: [2, 1] }
  },
  {
    $inc: { temperature: -1 }
  }
);
```

```
< {  
  averageTemperatureBefore: '-3.73',  
  averageTemperatureAfter: '-3.21',  
  temperatureChange: '0.51'  
}  
< {  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 368,  
  modifiedCount: 368,  
  upsertedCount: 0  
}  
test>
```