MongoDB Assignment 2

1. Create a database named `university` and a collection named `students`. Insert multiple student documents with fields: `name`, `age`, `department`, and `grades`.

   Creating database and collection

```
student_management> use university
switched to db university
university> db.createCollection("students")
{ ok: 1 }
university> show collections
students
university>
```

   Data insertion

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('67d6d185305fc2ddddcb0ce2'),
    '1': ObjectId('67d6d185305fc2ddddcb0ce3'),
    '2': ObjectId('67d6d185305fc2ddddcb0ce4')
  }
}
university> db.students.find()
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 20,
    department: 'computer science',
    grades: { math: 85, english: 92 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce3'),
    name: 'BOb',
    age: 21,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce4'),
    name: 'Charlie',
    age: 22,
    department: 'Mathematics',
    grades: { math: 95, statistics: 89 }
  }
]
university>
```

2. Write a query to display all students who are in the `Computer Science` department.

   db.students.find({department:"computer science"})

```
university> db.students.find({department:"computer science"})
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 20,
    department: 'computer science',
    grades: { math: 85, english: 92 }
  }
]
university>
```

3. Write a query to update the grades of a student named `Alice` by adding a new subject `programming` with a grade of 93.

```
db.students.updateOne(
  { name: "Alice" },
  { $set: { "grades.Programming": 93 } }
);
```

```
university> db.students.updateOne(
...     { name: "Alice" },
...     { $set: { "grades.Programming": 93 } }
... );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
university> db.students.find()
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 20,
    department: 'computer science',
    grades: { math: 85, english: 92, Programming: 93 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce3'),
    name: 'BOb',
    age: 21,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce4'),
    name: 'Charlie',
    age: 22,
    department: 'Mathematics',
    grades: { math: 95, statistics: 89 }
  }
]
university>
```

4. Write a query to increment the age of all students by 1.
   db.students.updateMany({},{$inc:{age:1}})

```
university>
(To exit, press Ctrl+C again or Ctrl+D or type .exit)
university> db.students.updateMany({},{$inc:{age:1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
university> db.students.find()
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 21,
    department: 'computer science',
    grades: { math: 85, english: 92, Programming: 93 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce3'),
    name: 'BOb',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce4'),
    name: 'Charlie',
    age: 23,
    department: 'Mathematics',
    grades: { math: 95, statistics: 89 }
  }
]
university>
```

5. Write a query to delete all students who are 23 years old.
   db.students.deleteMany({age:23})

```
university> db.students.deleteMany({age:23})
{ acknowledged: true, deletedCount: 1 }
university> db.students.find()
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 21,
    department: 'computer science',
    grades: { math: 85, english: 92, Programming: 93 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce3'),
    name: 'BOb',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  }
]
university>
```

6. Write a query to create an index on the `name` field of the `students` collection.

   db.students.createIndex({ name: 1 });

```
(To exit, press Ctrl+C again or Ctrl+D or type .exit)
university> db.students.createIndex({ name: 1 });
name_1
university> db.students.find()
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 21,
    department: 'computer science',
    grades: { math: 85, english: 92, Programming: 93 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce3'),
    name: 'BOb',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  }
]
university>
```

7. Write an aggregation query to group students by their department and calculate the average age in each department.

```
db.students.aggregate([
  {
    $group: {
      _id: "$department",
      averageAge: { $avg: "$age" }
    }}]);
```

```
university> db.students.aggregate([
...     {
...         $group: {
...             _id: "$department",
...             averageAge: { $avg: "$age" }
...         }
...     }
... ]);
[
  { _id: 'Physics', averageAge: 22 },
  { _id: 'computer science', averageAge: 21 }
]
university>
```

8. Write a query to find all students who have scored more than 90 in any subject.

```
db.students.find({
  $expr: {
    $gt: [
      { $max: { $objectToArray: "$grades" } },
      90
    ]
  }
});
```

```
university> db.students.find({
...     $expr: {
...         $gt: [
...             { $max: { $objectToArray: "$grades" } },
...             90
...         ]
...     }
... });
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 21,
    department: 'computer science',
    grades: { math: 85, english: 92, Programming: 93 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce3'),
    name: 'BOb',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  }
]
university>
```

9. Write a query to add a new field `graduated` set to `false` for all students who are in the `Mathematics` department.

```
db.students.updateMany(
  { department: "math" },
  { $set: { graduated: false } }
);
```

```
university> db.students.updateMany(
...     { department: "math" },
...     { $set: { graduated: false } }
... );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
university> db.students.find()
[
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce2'),
    name: 'Alice',
    age: 21,
    department: 'computer science',
    grades: { math: 85, english: 92, Programming: 93 }
  },
  {
    _id: ObjectId('67d6d185305fc2ddddcb0ce3'),
    name: 'BOb',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  },
  {
    '0': {
      name: 'Matt',
      age: 23,
      department: 'BCA',
      grades: { english: 90, physics: 92 }
    },
    _id: ObjectId('67d8064c305fc2ddddcb0ce5')
  }
]
university>
```

```
db.students.updateMany(
  { department: "Mathematics" },
  { $set: { graduated: false } }
);
```

10. How can you retrieve only the `name` and `department` fields for all students, excluding the `_id` field?

```
db.students.find(
   {},
   { name: 1, department: 1, _id: 0 });
```

```
university> db.students.find(
...     {},
...     { name: 1, department: 1, _id: 0 }
... );
[
  { name: 'Alice', department: 'computer science' },
  { name: 'BOb', department: 'Physics' },
  {}
]
university>
```