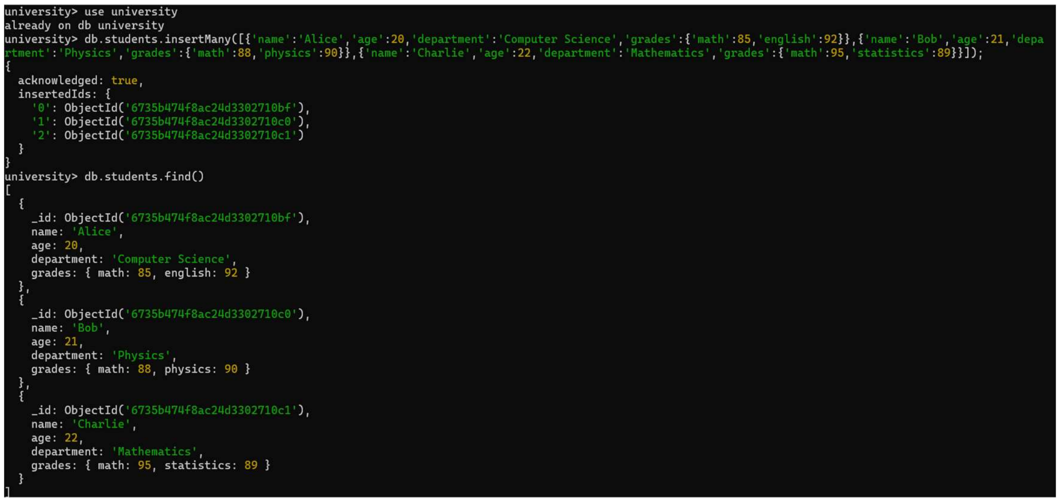
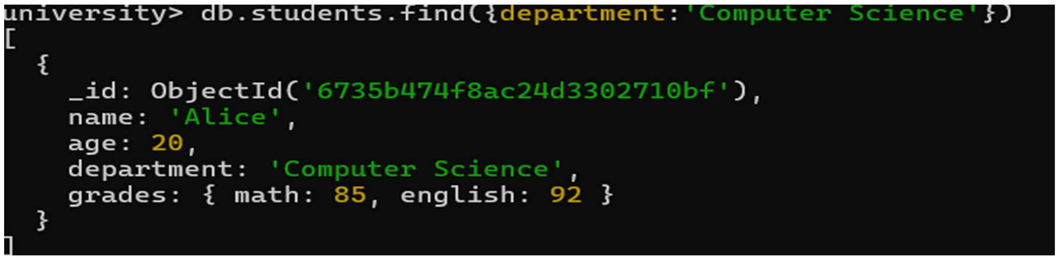
MongoDB Assignment – 2

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



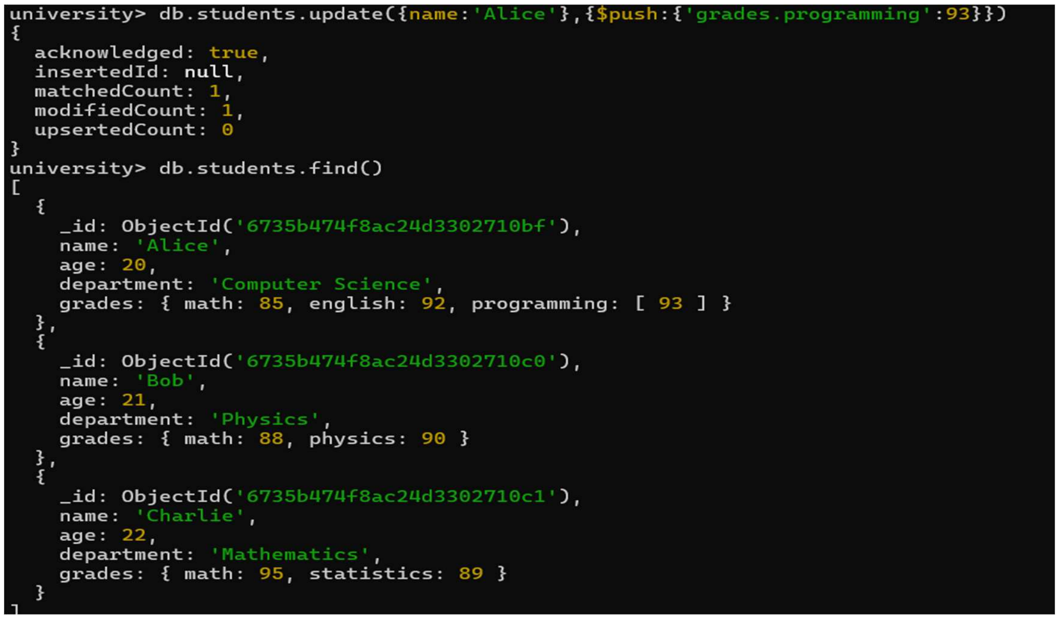
```
university> use university
already on db university
university> db.students.insertMany([{'name':'Alice','age':20,'department':'Computer Science','grades':{'math':85,'english':92}},{'name':'Bob','age':21,'depa
rtment':'Physics','grades':{'math':88,'physics':90}},{'name':'Charlie','age':22,'department':'Mathematics','grades':{'math':95,'statistics':89}}]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6735b474f8ac24d3302710bf'),
    '1': ObjectId('6735b474f8ac24d3302710c0'),
    '2': ObjectId('6735b474f8ac24d3302710c1')
  }
}
university> db.students.find()
[
  {
    _id: ObjectId('6735b474f8ac24d3302710bf'),
    name: 'Alice',
    age: 20,
    department: 'Computer Science',
    grades: { math: 85, english: 92 }
  },
  {
    _id: ObjectId('6735b474f8ac24d3302710c0'),
    name: 'Bob',
    age: 21,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  },
  {
    _id: ObjectId('6735b474f8ac24d3302710c1'),
    name: 'Charlie',
    age: 22,
    department: 'Mathematics',
    grades: { math: 95, statistics: 89 }
  }
]
```

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



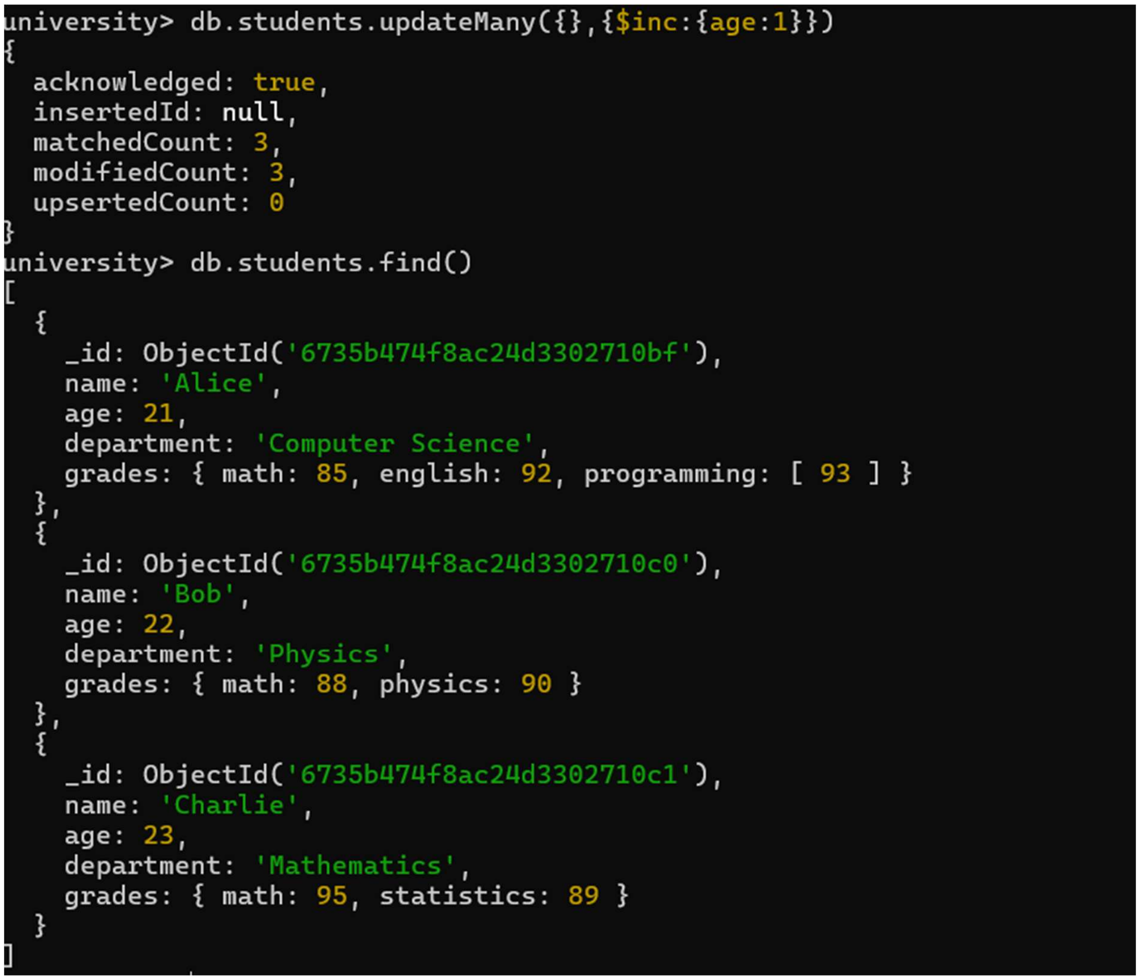
```
university> db.students.find({department:'Computer Science'})
[
  {
    _id: ObjectId('6735b474f8ac24d3302710bf'),
    name: 'Alice',
    age: 20,
    department: 'Computer Science',
    grades: { math: 85, english: 92 }
  }
]
```

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



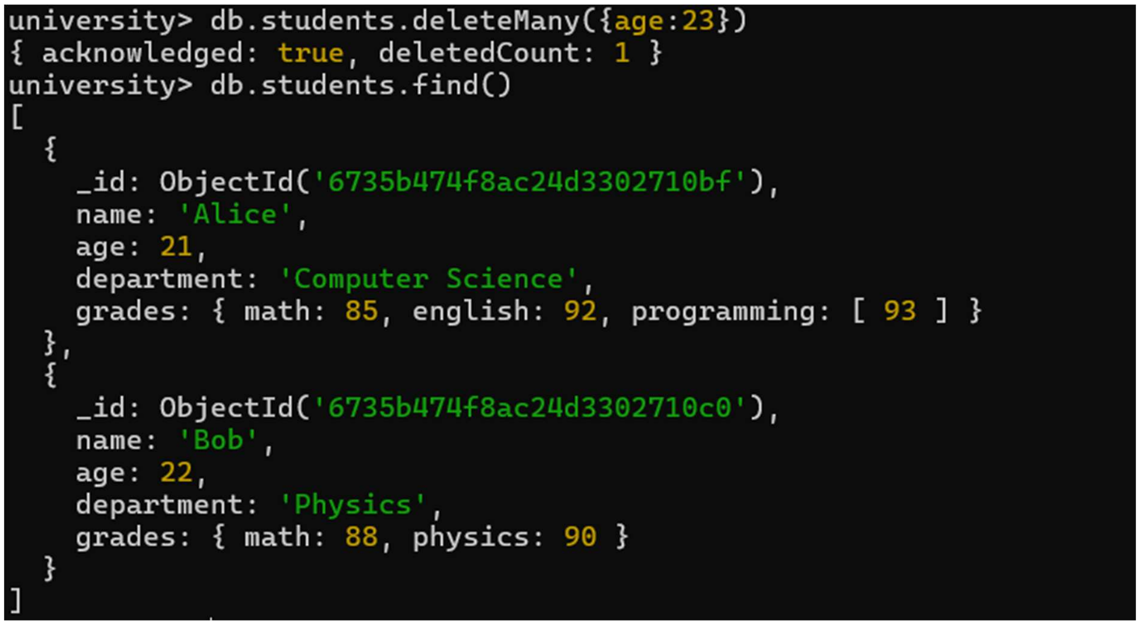
```
university> db.students.update({name:'Alice'},{$push:{'grades.programming':93}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
university> db.students.find()
[
  {
    _id: ObjectId('6735b474f8ac24d3302710bf'),
    name: 'Alice',
    age: 20,
    department: 'Computer Science',
    grades: { math: 85, english: 92, programming: [ 93 ] }
  },
  {
    _id: ObjectId('6735b474f8ac24d3302710c0'),
    name: 'Bob',
    age: 21,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  },
  {
    _id: ObjectId('6735b474f8ac24d3302710c1'),
    name: 'Charlie',
    age: 22,
    department: 'Mathematics',
    grades: { math: 95, statistics: 89 }
  }
]
```

4. Write a query to increment the age of all students by 1.



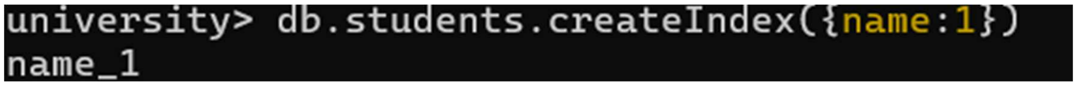
```
university> db.students.updateMany({},{$inc:{age:1}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 3,
  modifiedCount: 3,
  upsertedCount: 0
}
university> db.students.find()
[
  {
    _id: ObjectId('6735b474f8ac24d3302710bf'),
    name: 'Alice',
    age: 21,
    department: 'Computer Science',
    grades: { math: 85, english: 92, programming: [ 93 ] }
  },
  {
    _id: ObjectId('6735b474f8ac24d3302710c0'),
    name: 'Bob',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  },
  {
    _id: ObjectId('6735b474f8ac24d3302710c1'),
    name: 'Charlie',
    age: 23,
    department: 'Mathematics',
    grades: { math: 95, statistics: 89 }
  }
]
```

5. Write a query to delete all students who are 23 years old.



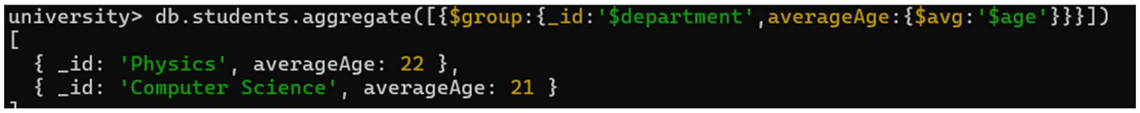
```
university> db.students.deleteMany({age:23})
{ acknowledged: true, deletedCount: 1 }
university> db.students.find()
[
  {
    _id: ObjectId('6735b474f8ac24d3302710bf'),
    name: 'Alice',
    age: 21,
    department: 'Computer Science',
    grades: { math: 85, english: 92, programming: [ 93 ] }
  },
  {
    _id: ObjectId('6735b474f8ac24d3302710c0'),
    name: 'Bob',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  }
]
```

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



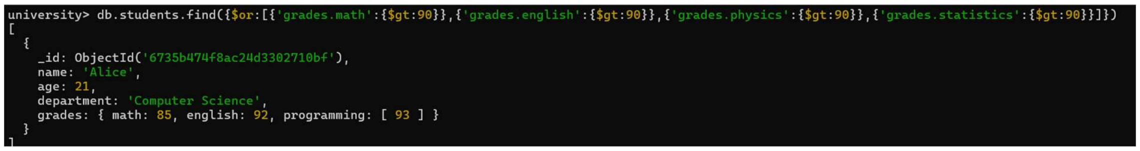
```
university> db.students.createIndex({name:1})
name_1
```

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



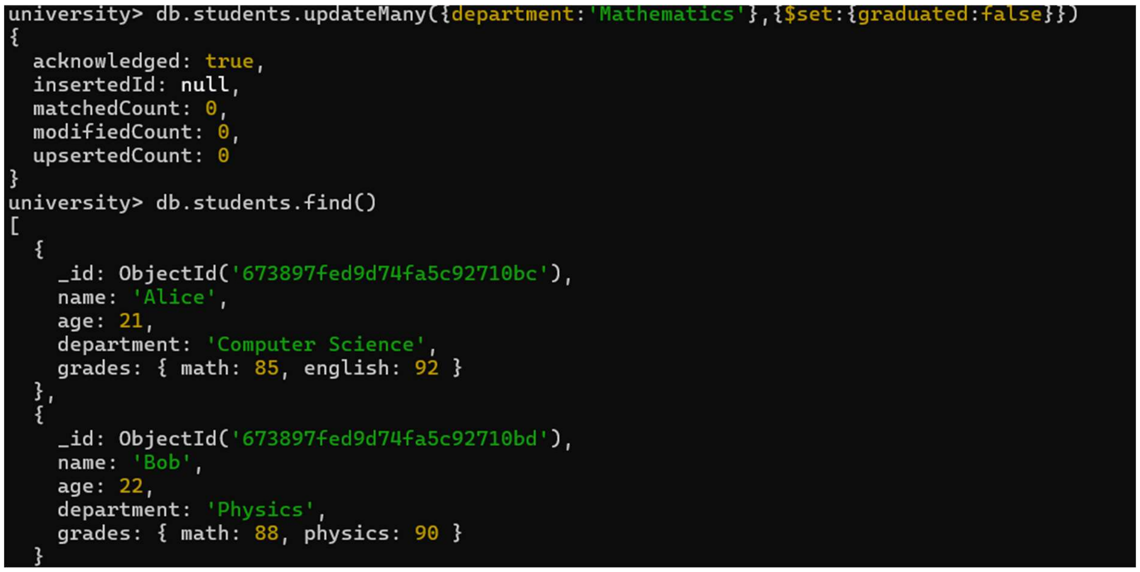
```
university> db.students.aggregate([{$group:{_id:'$department',averageAge:{$avg:'$age'}}}])
[
  { _id: 'Physics', averageAge: 22 },
  { _id: 'Computer Science', averageAge: 21 }
]
```

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



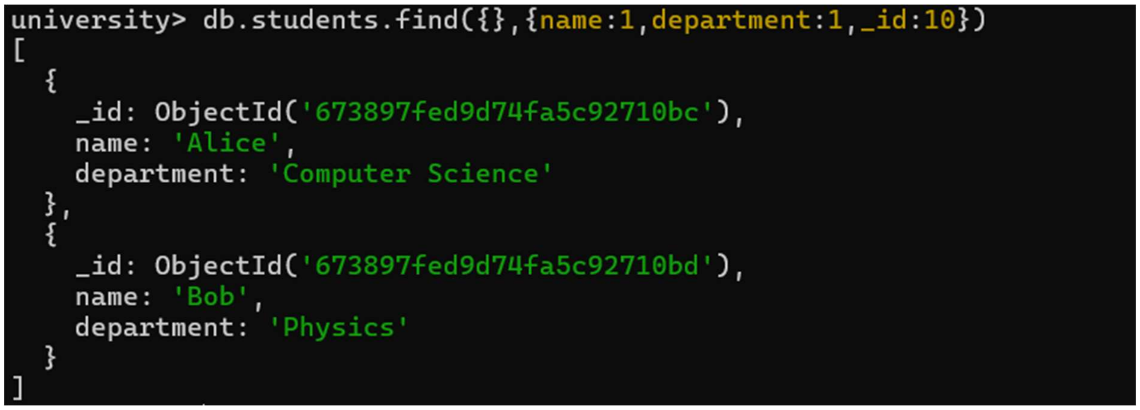
```
university> db.students.find({$or:[{'grades.math':{$gt:90}},{'grades.english':{$gt:90}},{'grades.physics':{$gt:90}},{'grades.statistics':{$gt:90}}]})
[
  {
    _id: ObjectId('6735b474f8ac24d3302710bf'),
    name: 'Alice',
    age: 21,
    department: 'Computer Science',
    grades: { math: 85, english: 92, programming: [ 93 ] }
  }
]
```

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



```
university> db.students.updateMany({department:'Mathematics'},{$set:{graduated:false}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
university> db.students.find()
[
  {
    _id: ObjectId('673897fed9d74fa5c92710bc'),
    name: 'Alice',
    age: 21,
    department: 'Computer Science',
    grades: { math: 85, english: 92 }
  },
  {
    _id: ObjectId('673897fed9d74fa5c92710bd'),
    name: 'Bob',
    age: 22,
    department: 'Physics',
    grades: { math: 88, physics: 90 }
  }
```

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



```
university> db.students.find({},{name:1,department:1,_id:10})
[
  {
    _id: ObjectId('673897fed9d74fa5c92710bc'),
    name: 'Alice',
    department: 'Computer Science'
  },
  {
    _id: ObjectId('673897fed9d74fa5c92710bd'),
    name: 'Bob',
    department: 'Physics'
  }
]
```