MongoDB - 1

1. Create a new database called student_management.

use student_management

2. Create a collection called students in the student_management database.

db.createCollection(“students”)

3. Insert at least five student records into the students collection. Each record should have the following fields:

- student_id (integer)
- name (string)
- age (integer)
- department (string)
- courses (array of strings)
- grade (string)

db.student.insertMany([

{student_id:101,

name:'Luv Goel',

age:20,

department:'Data Science',

courses:["C++","Python"],

grade:'A'},

{student_id:102,

name:'Ravi Sankar',

age:22,

department:'Computer Science',

courses:['PowerBI','MySQL'],

grade:'C'},

{student_id:103,

name:'Jaya Prakash',

age:21,

department:'Computer Applications',

courses:['MongoDB','React'],

grade:'A'},

{student_id:104,

name:'Shyam Dubey',

age:19,

department:'Computer Science',

courses:['Big Data', 'Cloud Computing'],

grade:'B'},

{student_id:105,

name:'Nikhil Raj',

age:25,

department:'Artificial Intelligence',

courses:['Database Systems'],

grade:'A'}])

```
mycompiler_mongodb> ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ..
 acknowledged: true,
 insertedIds: {
   '0': ObjectId('67cf0ca7edafd945de6b128c'),
   '1': ObjectId('67cf0ca7edafd945de6b128d'),
   '2': ObjectId('67cf0ca7edafd945de6b128e'),
   '3': ObjectId('67cf0ca7edafd945de6b128f'),
   '4': ObjectId('67cf0ca7edafd945de6b1290')
 }
}
mycompiler_mongodb>

[Execution complete with exit code 0]
```

4. Query the Collection:

Write queries to perform the following tasks:

- Retrieve all students who are in the "Computer Science" department.

db.student.find({department:'Computer Science'})

```
mycompiler_mongodb> [
  {
    _id: ObjectId('67cf0d2d9d4d63ec6a6b128d'),
    student_id: 102,
    name: 'Ravi Sankar',
    age: 22,
    department: 'Computer Science',
    courses: [ 'PowerBI', 'MySQL' ],
    grade: 'C'
  },
  {
    _id: ObjectId('67cf0d2d9d4d63ec6a6b128f'),
    student_id: 104,
    name: 'Shyam Dubey',
    age: 19,
    department: 'Computer Science',
    courses: [ 'Big Data', 'Cloud Computing' ],
    grade: 'B'
  }
]
mycompiler_mongodb>

[Execution complete with exit code 0]
```

- Retrieve students who have an age greater than 21.

db.student.find({age:{$gt:21}})

```
mycompiler_mongodb> [
  {
    _id: ObjectId('67cf0eb58aaf1a937c6b128d'),
    student_id: 102,
    name: 'Ravi Sankar',
    age: 22,
    department: 'Computer Science',
    courses: [ 'PowerBI', 'MySQL' ],
    grade: 'C'
  },
  {
    _id: ObjectId('67cf0eb58aaf1a937c6b1290'),
    student_id: 105,
    name: 'Nikhil Raj',
    age: 25,
    department: 'Artificial Intelligence',
    courses: [ 'Database Systems' ],
    grade: 'A'
  }
]
mycompiler_mongodb>

[Execution complete with exit code 0]
```

- Retrieve students who are taking the "Database Systems" course.

db.student.find({courses:'Database Systems'})

```
mycompiler_mongodb> [
  {
    _id: ObjectId('67cf0fa5ea643e157e6b1290'),
    student_id: 105,
    name: 'Nikhil Raj',
    age: 25,
    department: 'Artificial Intelligence',
    courses: [ 'Database Systems' ],
    grade: 'A'
  }
]
mycompiler_mongodb>

[Execution complete with exit code 0]
```

- Retrieve students with a grade of "A".

db.student.find({grade: 'A' })

```
mycompiler_mongodb> [
  {
    _id: ObjectId('67cf10001afc9f228d6b128c'),
    student_id: 101,
    name: 'Luv Goel',
    age: 20,
    department: 'Data Science',
    courses: [ 'C++', 'Python' ],
    grade: 'A'
  },
  {
    _id: ObjectId('67cf10001afc9f228d6b128e'),
    student_id: 103,
    name: 'Jaya Prakash',
    age: 21,
    department: 'Computer Applications',
    courses: [ 'MongoDB', 'React' ],
    grade: 'A'
  },
  {
    _id: ObjectId('67cf10001afc9f228d6b1290'),
    student_id: 105,
    name: 'Nikhil Raj',
    age: 25,
    department: 'Artificial Intelligence',
    courses: [ 'Database Systems' ],
    grade: 'A'
  }
]
mycompiler_mongodb>

[Execution complete with exit code 0]
```

5. Update Documents:

- Update the age of a student with student_id 101 to 21.

db.student.updateOne({student_id:101},{$set:{age:21}})

db.student.find({student_id:101})

```
mycompiler_mongodb> {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
mycompiler_mongodb> [
  {
    _id: ObjectId('67cf117b6d516e8b426b128c'),
    student_id: 101,
    name: 'Luv Goel',
    age: 21,
    department: 'Data Science',
    courses: [ 'C++', 'Python' ],
    grade: 'A'
  }
]
mycompiler_mongodb>

[Execution complete with exit code 0]
```

- Add a new course, "Machine Learning", to the courses array for students in the "Computer Science" department.

db.student.updateMany({department:'Computer Science'},{$push:{courses:

'Machine Learning'}})

db.student.find({department:'Computer Science'})

```
mycompiler_mongodb> {
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
mycompiler_mongodb> [
  {
    _id: ObjectId('67cf18762a6ca786076b128d'),
    student_id: 102,
    name: 'Ravi Sankar',
    age: 22,
    department: 'Computer Science',
    courses: [ 'PowerBI', 'MySQL', 'Machine Learning' ],
    grade: 'C'
  },
  {
    _id: ObjectId('67cf18762a6ca786076b128f'),
    student_id: 104,
    name: 'Shyam Dubey',
    age: 19,
    department: 'Computer Science',
    courses: [ 'Big Data', 'Cloud Computing', 'Machine Learning' ],
    grade: 'B'
  }
]
mycompiler_mongodb>

[Execution complete with exit code 0]
```

6. Delete Documents:

- Delete a student record with student_id 105.

db.student.deleteOne({student_id:105})

```
mycompiler_mongodb> { acknowledged: true, deletedCount: 1 }
mycompiler_mongodb>

[Execution complete with exit code 0]
```

- Delete all students who have a grade lower than "C".

db.student.deleteMany({grade:{$lt:'C'}})

```
mycompiler_mongodb> { acknowledged: true, deletedCount: 4 }
mycompiler_mongodb>

[Execution complete with exit code 0]
```