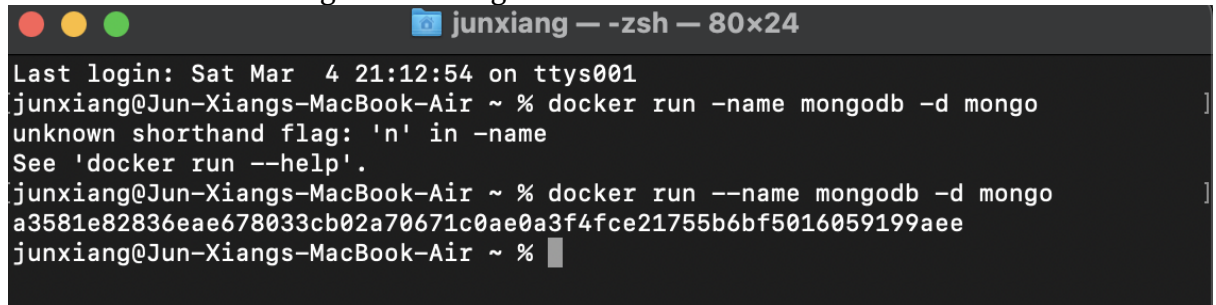


Question 1: Activate/run Docker and install/pull MongoDB in Docker. You are suggested to use command-line tools, e.g., PowerShell in Windows.
(Although not the focus of this module, you are expected to be able to develop efficiently in Linux.)

Command used:

1. `docker run --name mongodb -d mongo`

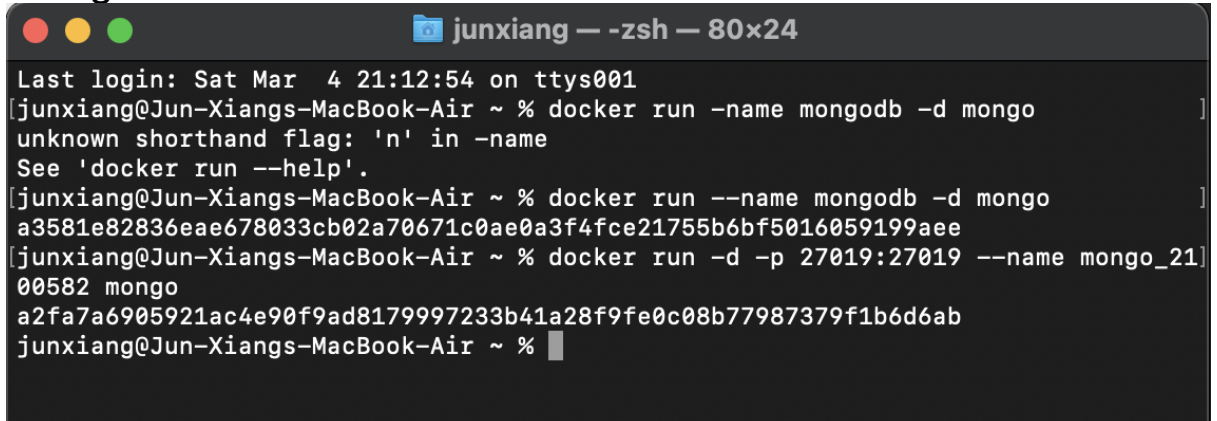


```
junxiang — zsh — 80x24
Last login: Sat Mar  4 21:12:54 on ttys001
junxiang@Jun-Xiangs-MacBook-Air ~ % docker run --name mongodb -d mongo
unknown shorthand flag: 'n' in --name
See 'docker run --help'.
junxiang@Jun-Xiangs-MacBook-Air ~ % docker run --name mongodb -d mongo
a3581e82836eae678033cb02a70671c0ae0a3f4fce21755b6bf5016059199aee
junxiang@Jun-Xiangs-MacBook-Air ~ %
```

Question 2: Start a Docker container listening on port 27019 (default MongoDB port). Specify the name as mongo_<student_id>, e.g., mongo_2101234.

Commands used:

1. docker run -d -p 27019:27019 --name mongo_2100582 mongo

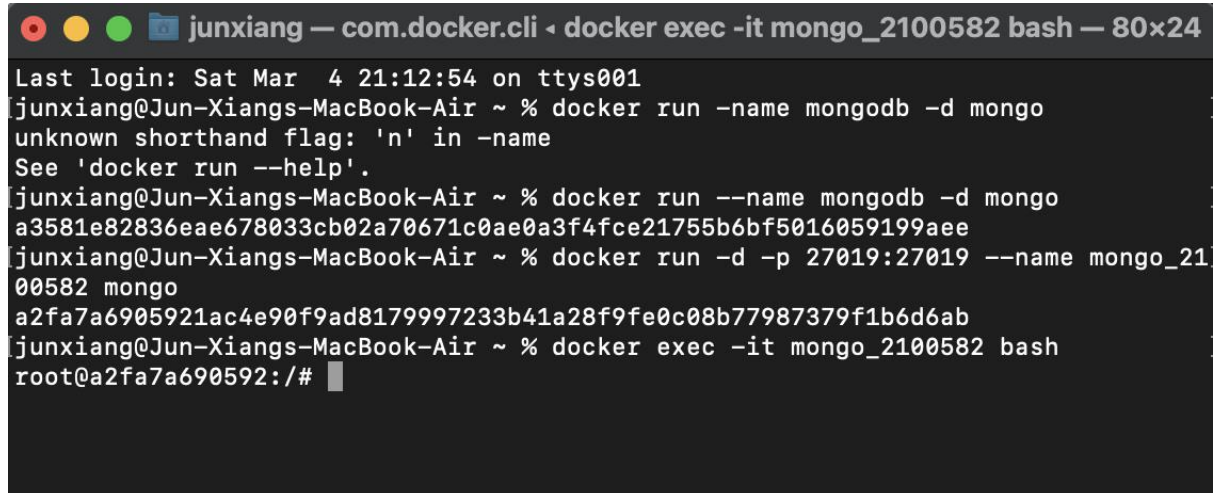
A terminal window titled 'junxiang — zsh — 80x24' with standard macOS window controls. The terminal shows the following sequence of commands and outputs:

```
Last login: Sat Mar  4 21:12:54 on ttys001
[junxiang@Jun-Xiangs-MacBook-Air ~ % docker run -name mongodb -d mongo
unknown shorthand flag: 'n' in -name
See 'docker run --help'.
[junxiang@Jun-Xiangs-MacBook-Air ~ % docker run --name mongodb -d mongo
a3581e82836eae678033cb02a70671c0ae0a3f4fce21755b6bf5016059199aee
[junxiang@Jun-Xiangs-MacBook-Air ~ % docker run -d -p 27019:27019 --name mongo_21]
00582 mongo
a2fa7a6905921ac4e90f9ad8179997233b41a28f9fe0c08b77987379f1b6d6ab
junxiang@Jun-Xiangs-MacBook-Air ~ %
```

Question 3: Enter the docker container with MongoDB just installed. You may see the account (root) is no longer the one for your computer.

Commands used:

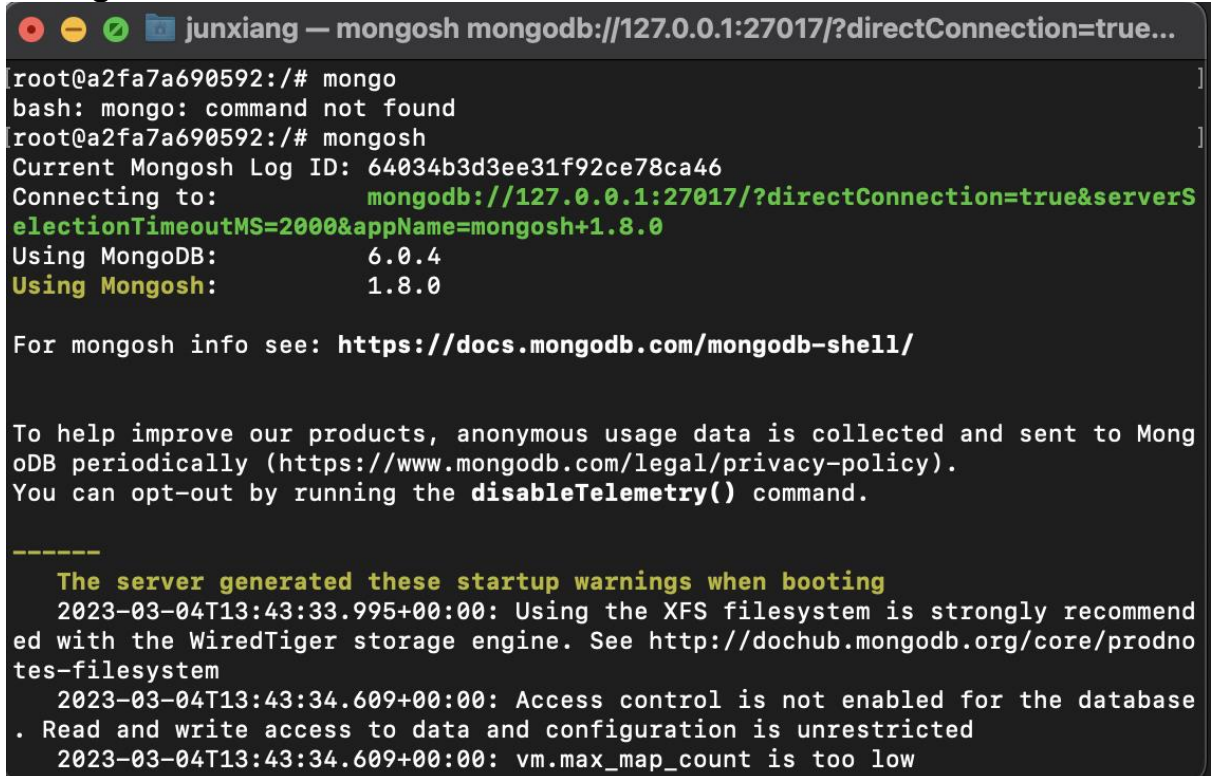
1. `docker exec -it mongo_2100582 bash`

A terminal window titled 'junxiang — com.docker.cli • docker exec -it mongo_2100582 bash — 80x24'. The terminal shows the following sequence of commands and output:
Last login: Sat Mar 4 21:12:54 on ttys001
junxiang@Jun-Xiangs-MacBook-Air ~ % docker run -name mongodb -d mongo
unknown shorthand flag: 'n' in -name
See 'docker run --help'.
junxiang@Jun-Xiangs-MacBook-Air ~ % docker run --name mongodb -d mongo
a3581e82836eae678033cb02a70671c0ae0a3f4fce21755b6bf5016059199aee
junxiang@Jun-Xiangs-MacBook-Air ~ % docker run -d -p 27019:27019 --name mongo_2100582 mongo
a2fa7a6905921ac4e90f9ad8179997233b41a28f9fe0c08b77987379f1b6d6ab
junxiang@Jun-Xiangs-MacBook-Air ~ % docker exec -it mongo_2100582 bash
root@a2fa7a690592:/#

Question 4: Enter MongoDB by typing mongo. You should be in the MongoDB environment already.

Commands used:

1. mongosh

A terminal window titled 'junxiang — mongosh mongodb://127.0.0.1:27017/?directConnection=true...' shows a user attempting to connect to MongoDB. The user first types 'mongo', which fails with 'bash: mongo: command not found'. Then, the user types 'mongosh', which successfully connects to the MongoDB shell. The terminal displays the current log ID, the connection string, and the versions of MongoDB (6.0.4) and mongosh (1.8.0). It also provides a link for more information and a notice about telemetry data collection. Finally, it shows startup warnings from the MongoDB server, including a recommendation to use the WiredTiger storage engine and a warning about access control being disabled.

```
root@a2fa7a690592:/# mongo
bash: mongo: command not found
root@a2fa7a690592:/# mongosh
Current Mongosh Log ID: 64034b3d3ee31f92ce78ca46
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverS
electionTimeoutMS=2000&appName=mongosh+1.8.0
Using MongoDB:      6.0.4
Using Mongosh:      1.8.0

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

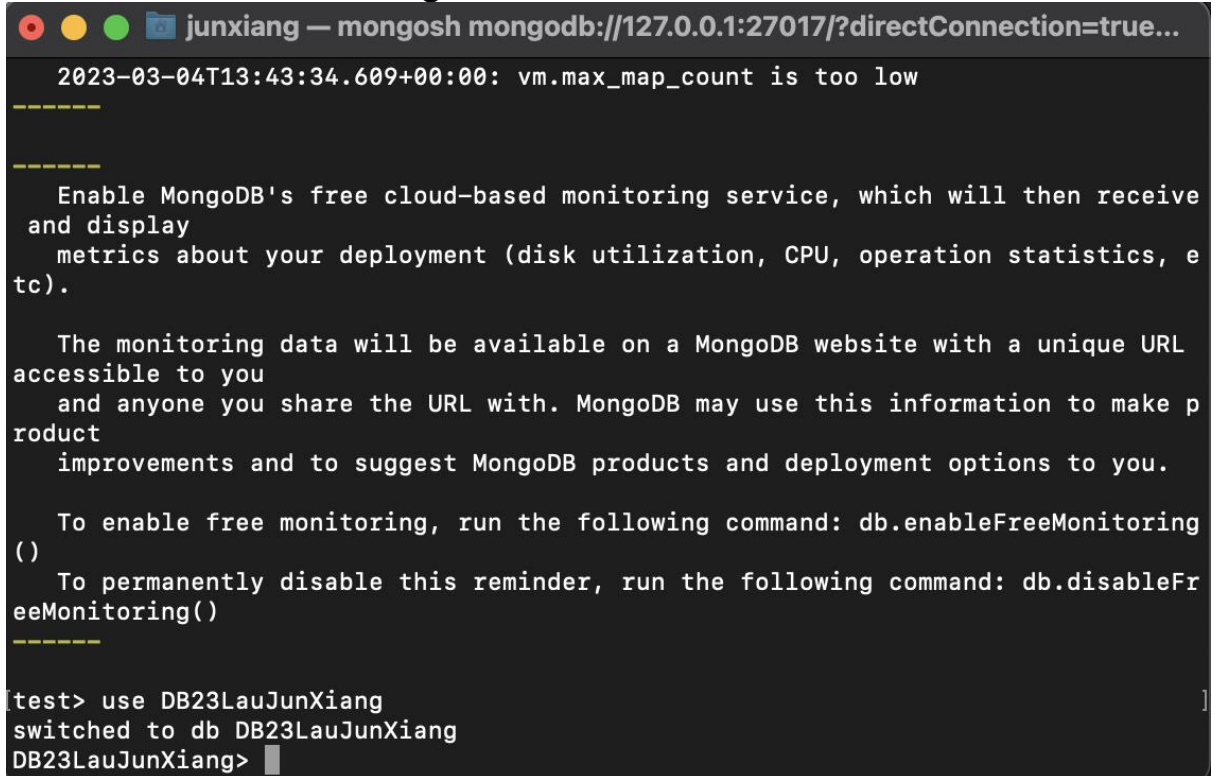
To help improve our products, anonymous usage data is collected and sent to Mong
oDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

-----
The server generated these startup warnings when booting
2023-03-04T13:43:33.995+00:00: Using the XFS filesystem is strongly recommend
ed with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodno
tes-filesystem
2023-03-04T13:43:34.609+00:00: Access control is not enabled for the database
. Read and write access to data and configuration is unrestricted
2023-03-04T13:43:34.609+00:00: vm.max_map_count is too low
```

Question 5: Create a database called DB23<student_name>, e.g., DB23Wei. (Do you have to check if it exists in MongoDB?)

Command Used:

1. use DB23LauJunXiang

A screenshot of a terminal window titled 'junxiang — mongosh mongodb://127.0.0.1:27017/?directConnection=true...'. The terminal shows a warning message: '2023-03-04T13:43:34.609+00:00: vm.max_map_count is too low'. Below this, there is a block of text explaining MongoDB's free cloud-based monitoring service. At the bottom, the command 'test> use DB23LauJunXiang' is entered, and the output is 'switched to db DB23LauJunXiang' followed by a prompt 'DB23LauJunXiang>'.

```
junxiang — mongosh mongodb://127.0.0.1:27017/?directConnection=true...
2023-03-04T13:43:34.609+00:00: vm.max_map_count is too low

-----

Enable MongoDB's free cloud-based monitoring service, which will then receive
and display
metrics about your deployment (disk utilization, CPU, operation statistics, e
tc).

The monitoring data will be available on a MongoDB website with a unique URL
accessible to you
and anyone you share the URL with. MongoDB may use this information to make p
roduct
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring
()
To permanently disable this reminder, run the following command: db.disableFr
eeMonitoring()

-----

test> use DB23LauJunXiang
switched to db DB23LauJunXiang
DB23LauJunXiang>
```

Question 6: Create a collection Students.

Command used:

1. db.createCollection("Students")

```
[test> use DB23LauJunXiang  
switched to db DB23LauJunXiang  
[DB23LauJunXiang> db.createCollection("Students")  
{ ok: 1 }  
DB23LauJunXiang> ]
```

Question 7: Insert the below student data into the Students collection. Each row corresponds to three attributes, namely sname, syear, and major.

Iron Man 2019 engineering

Deadpool 2019 biology

Wolverine 2020 physics

Hulk 2020 physics

Thor 2021 physics

Rocket 2021 engineering

<student_name> 2021 engineering

Command used:

1. db.Students.insertOne({sname: "Iron Man", syear: 2019, major: "engineering"})
2. db.Students.insertOne({sname: "Deadpool", syear: 2019, major: "biology"})
3. db.Students.insertOne({sname: "Wolverine", syear: 2020, major: "physics"})
4. db.Students.insertOne({sname: "Hulk", syear: 2020, major: "physics"})
5. db.Students.insertOne({sname: "Thor", syear: 2021, major: "physics"})
6. db.Students.insertOne({sname: "Rocket", syear: 2021, major: "engineering"})
7. db.Students.insertOne({sname: "Lau Jun Xiang", syear: 2021, major: "engineering"})

```
DB23LauJunXiang> db.Students.insertOne({sname: "Iron Man", syear: 2019, major: "engineering"})
{
  acknowledged: true,
  insertedId: ObjectId("640353f117b9051cacba145b")
}
DB23LauJunXiang> db.Students.insertOne({sname: "Deadpool", syear: 2019, major: "biology"})
{
  acknowledged: true,
  insertedId: ObjectId("640353f117b9051cacba145c")
}
DB23LauJunXiang> db.Students.insertOne({sname: "Wolverine", syear: 2020, major: "physics"})
{
  acknowledged: true,
  insertedId: ObjectId("640353f117b9051cacba145d")
}
DB23LauJunXiang> db.Students.insertOne({sname: "Hulk", syear: 2020, major: "physics"})
{
  acknowledged: true,
  insertedId: ObjectId("640353f117b9051cacba145e")
}
DB23LauJunXiang> db.Students.insertOne({sname: "Thor", syear: 2021, major: "physics"})
{
  acknowledged: true,
  insertedId: ObjectId("640353f117b9051cacba145f")
}
DB23LauJunXiang> db.Students.insertOne({sname: "Rocket", syear: 2021, major: "engineering"})
{
  acknowledged: true,
  insertedId: ObjectId("640353f117b9051cacba1460")
}
DB23LauJunXiang> db.Students.insertOne({sname: "Lau Jun Xiang", syear: 2021, major: "engineering"})
{
  acknowledged: true,
  insertedId: ObjectId("640353f117b9051cacba1461")
}
DB23LauJunXiang> █
```

Question 8: Display data in the Students collection. Observe if we get new data besides what we just inserted.

Commands used:

1. db.students.find()

```
[DB23LauJunXiang> db.Students.find()
[
  {
    _id: ObjectId("640353f117b9051cacba145b"),
    sname: 'Iron Man',
    syear: 2019,
    major: 'engineering'
  },
  {
    _id: ObjectId("640353f117b9051cacba145c"),
    sname: 'Deadpool',
    syear: 2019,
    major: 'biology'
  },
  {
    _id: ObjectId("640353f117b9051cacba145d"),
    sname: 'Wolverine',
    syear: 2020,
    major: 'physics'
  },
  {
    _id: ObjectId("640353f117b9051cacba145e"),
    sname: 'Hulk',
    syear: 2020,
    major: 'physics'
  },
  {
    _id: ObjectId("640353f117b9051cacba145f"),
    sname: 'Thor',
    syear: 2021,
    major: 'physics'
  },
  {
    _id: ObjectId("640353f117b9051cacba1460"),
    sname: 'Rocket',
    syear: 2021,
    major: 'engineering'
  },
  {
    _id: ObjectId("640353f117b9051cacba1461"),
    sname: 'Lau Jun Xiang',
    syear: 2021,
    major: 'engineering'
  }
]
DB23LauJunXiang> █
```

Question 9: Deadpool wants to switch major from biology to art, please update the collection accordingly.

Command used:

1. `db.Students.updateOne({sname:"Deadpool"},{$set: {major:"art"}})`
`db.Students.find({sname:"Deadpool"})`

```
[DB23LauJunXiang> db.Students.updateOne({sname:"Deadpool"},{$set: {major:"art"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
[DB23LauJunXiang> db.Students.find({sname:"Deadpool"})
[
  {
    _id: ObjectId("640353f117b9051cacba145c"),
    sname: 'Deadpool',
    syear: 2019,
    major: 'art'
  }
]
DB23LauJunXiang> █
```

Question 10: Add a new field campus with the default value "SIT@NYP" for all students. (You may display the collection to verify. If the update did not apply to all students, why and how to solve it?)

Commands used:

1. `db.Students.updateMany({}, {$set: {campus: "SIT@NYP"}})`

```
[DB23LauJunXiang> db.Students.updateMany({}, {$set: {campus: "SIT@NYP"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 7,
  modifiedCount: 7,
  upsertedCount: 0
}
DB23LauJunXiang> █
```

2. `db.Students.find()`

```

[DB23LauJunXiang> db.Students.find()
[
  {
    _id: ObjectId("640353f117b9051cacba145b"),
    sname: 'Iron Man',
    syear: 2019,
    major: 'engineering',
    campus: 'SIT@NYP'
  },
  {
    _id: ObjectId("640353f117b9051cacba145c"),
    sname: 'Deadpool',
    syear: 2019,
    major: 'art',
    campus: 'SIT@NYP'
  },
  {
    _id: ObjectId("640353f117b9051cacba145d"),
    sname: 'Wolverine',
    syear: 2020,
    major: 'physics',
    campus: 'SIT@NYP'
  },
  {
    _id: ObjectId("640353f117b9051cacba145e"),
    sname: 'Hulk',
    syear: 2020,
    major: 'physics',
    campus: 'SIT@NYP'
  },
  {
    _id: ObjectId("640353f117b9051cacba145f"),
    sname: 'Thor',
    syear: 2021,
    major: 'physics',
    campus: 'SIT@NYP'
  },
  {
    _id: ObjectId("640353f117b9051cacba1460"),
    sname: 'Rocket',
    syear: 2021,
    major: 'engineering',
    campus: 'SIT@NYP'
  },
  {
    _id: ObjectId("640353f117b9051cacba1461"),
    sname: 'Lau Jun Xiang',
    syear: 2021,
    major: 'engineering',
    campus: 'SIT@NYP'
  }
]
DB23LauJunXiang> █

```

3. If it did not update, it probably is because campus has already existed in one of them. We can solve it by using `db.Students.updateMany({}, {$set: {campus: "SIT@NYP"}}, {upsert:true})`. The command will insert if it does not exist or update if it exists.

Question 11: Add a new field GPA with a default value of 4.0 for all students

Commands used:

1. `db.Students.updateMany({}, {$set: {GPA:4.0}})`

```
[DB23LauJunXiang> db.Students.updateMany( {} , {$set: {GPA:4.0}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 7,
  modifiedCount: 7,
  upsertedCount: 0
}
DB23LauJunXiang> █
```

2. db.Students.find()

```
sname: 'Iron Man',
syear: 2019,
major: 'engineering',
campus: 'SIT@NYP',
GPA: 4
},
{
  _id: ObjectId("640353f117b9051cacba145c"),
  sname: 'Deadpool',
  syear: 2019,
  major: 'art',
  campus: 'SIT@NYP',
  GPA: 4
},
{
  _id: ObjectId("640353f117b9051cacba145d"),
  sname: 'Wolverine',
  syear: 2020,
  major: 'physics',
  campus: 'SIT@NYP',
  GPA: 4
},
{
  _id: ObjectId("640353f117b9051cacba145e"),
  sname: 'Hulk',
  syear: 2020,
  major: 'physics',
  campus: 'SIT@NYP',
  GPA: 4
},
{
  _id: ObjectId("640353f117b9051cacba145f"),
  sname: 'Thor',
  syear: 2021,
  major: 'physics',
  campus: 'SIT@NYP',
  GPA: 4
},
{
  _id: ObjectId("640353f117b9051cacba1460"),
  sname: 'Rocket',
  syear: 2021,
  major: 'engineering',
  campus: 'SIT@NYP',
  GPA: 4
},
{
  _id: ObjectId("640353f117b9051cacba1461"),
  sname: 'Lau Jun Xiang',
  syear: 2021,
  major: 'engineering',
  campus: 'SIT@NYP',
  GPA: 4
}
```


Question 12:

Commands used:

1. `db.Students.updateOne({sname: "Rocket"}, {$set: {GPA:5.0}})`

`db.Students.find({sname:"Rocket"})`

```
DB23LauJunXiang> db.Students.updateOne({sname: "Rocket"} , {$set: {GPA:5.0}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
DB23LauJunXiang> db.Students.find({sname: "Rocket"})
[
  {
    _id: ObjectId("640353f117b9051cacba1460"),
    sname: 'Rocket',
    year: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 5
  }
]
DB23LauJunXiang> 
```

Question 13: Find the students whose name has a character "a".

Commands used:

1. `db.Students.find({sname: /a/})`

```
[DB23LauJunXiang> db.Students.find({sname: /a/})
[
  {
    _id: ObjectId("640353f117b9051cacba145b"),
    sname: 'Iron Man',
    syear: 2019,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 4
  },
  {
    _id: ObjectId("640353f117b9051cacba145c"),
    sname: 'Deadpool',
    syear: 2019,
    major: 'art',
    campus: 'SIT@NYP',
    GPA: 4
  },
  {
    _id: ObjectId("640353f117b9051cacba1461"),
    sname: 'Lau Jun Xiang',
    syear: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 4
  }
]
DB23LauJunXiang> 
```

Question 14: Find engineering students.

Commands used:

1. `db.Students.find({major: "engineering"})`

```
[DB23LauJunXiang> db.Students.find({major: "engineering"})
[
  {
    _id: ObjectId("640353f117b9051cacba145b"),
    sname: 'Iron Man',
    syear: 2019,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 4
  },
  {
    _id: ObjectId("640353f117b9051cacba1460"),
    sname: 'Rocket',
    syear: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 5
  },
  {
    _id: ObjectId("640353f117b9051cacba1461"),
    sname: 'Lau Jun Xiang',
    syear: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 4
  }
]
DB23LauJunXiang> █
```

Question 15: Return the number of non-physics students

Command used:

1. `db.Students.countDocuments({major: {$ne: "physics"}})`

```
DB23LauJunXiang> db.Students.countDocuments({major: {$ne: "physics"}})
4
DB23LauJunXiang> █
```

Question 16: Find the students whose syear > 2020 and major in engineering

Command used:

1. `db.Students.find({syear: {$gt: 2020}, major: "engineering"})`

```
DB23LauJunXiang> db.Students.find({syear: {$gt: 2020}, major: "engineering"})
[
  {
    _id: ObjectId("640353f117b9051cacba1460"),
    sname: 'Rocket',
    syear: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 5
  },
  {
    _id: ObjectId("640353f117b9051cacba1461"),
    sname: 'Lau Jun Xiang',
    syear: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 4
  }
]
DB23LauJunXiang> █
```

Question 17:

Commands used:

1. `db.Students.find().sort({year: -1})`

```
DB23LauJunXiang> db.Students.find().sort({year: -1})
[
  {
    _id: ObjectId("640353f117b9051cacba145f"),
    sname: 'Thor',
    year: 2021,
    major: 'physics',
    campus: 'SIT@NYP',
    GPA: 4
  },
  {
    _id: ObjectId("640353f117b9051cacba1460"),
    sname: 'Rocket',
    year: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 5
  },
  {
    _id: ObjectId("640353f117b9051cacba1461"),
    sname: 'Lau Jun Xiang',
    year: 2021,
    major: 'engineering',
    campus: 'SIT@NYP',
    GPA: 4
  },
  {
    _id: ObjectId("640353f117b9051cacba145d"),
    sname: 'Wolverine',
    year: 2020,
    major: 'physics',
    campus: 'SIT@NYP',
    GPA: 4
  },
  {
    _id: ObjectId("640353f117b9051cacba145e"),
    sname: 'Hulk',
    year: 2020,
    major: 'physics',
    campus: 'SIT@NYP',
    GPA: 4
  },
]
```

```
{
  _id: ObjectId("640353f117b9051cacba145b"),
  sname: 'Iron Man',
  year: 2019,
  major: 'engineering',
  campus: 'SIT@NYP',
  GPA: 4
},
{
  _id: ObjectId("640353f117b9051cacba145c"),
  sname: 'Deadpool',
  year: 2019,
  major: 'art',
  campus: 'SIT@NYP',
  GPA: 4
}
]
DB23LauJunXiang>
```

Question 18: Display the number of students for each major and display them in descending order of the number of students.

Command used:

```
1. db.Students.aggregate([
  {$group: {_id: "$major", count: {$sum: 1}}},
  {$sort: {count: -1}}
])
```

```
DB23LauJunXiang> db.Students.aggregate([
...   {$group: {_id: "$major", count: {$sum: 1}}},
...   {$sort: {count: -1}}
... ])
[
  { _id: 'engineering', count: 3 },
  { _id: 'physics', count: 3 },
  { _id: 'art', count: 1 }
]
DB23LauJunXiang>
```