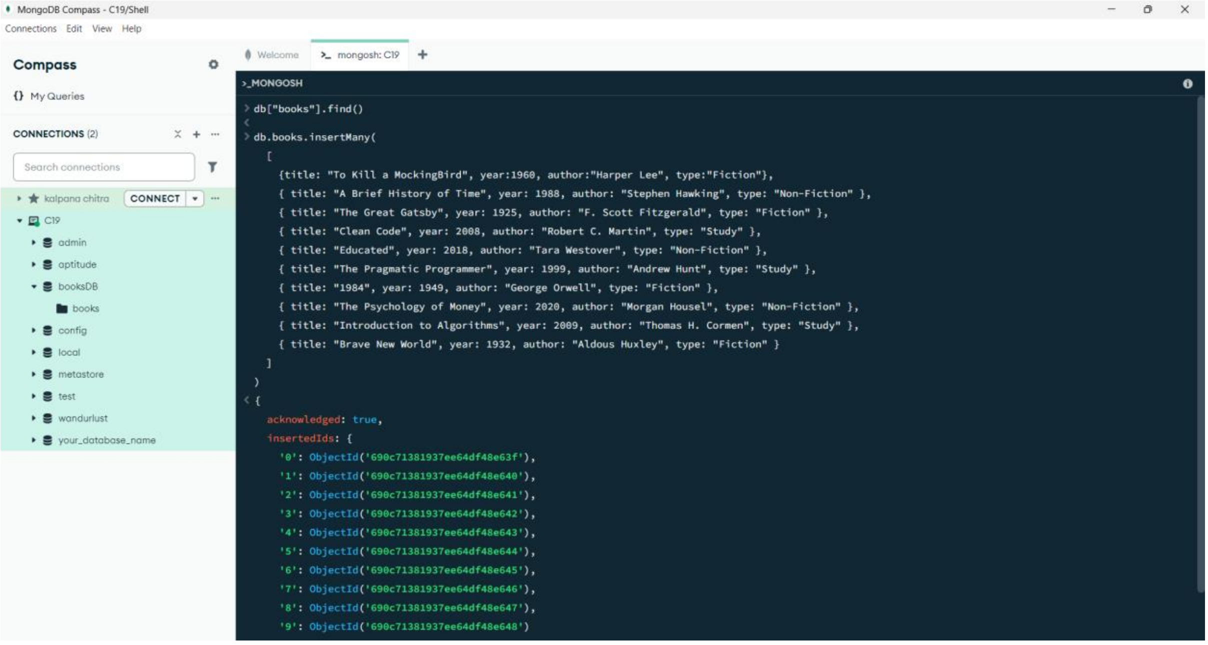
EXPERIMENT 6

Name:- Omkar Kore **Class**: Btech C

Roll Number:- BTC19 **Batch**: C1

MongoDB Queries

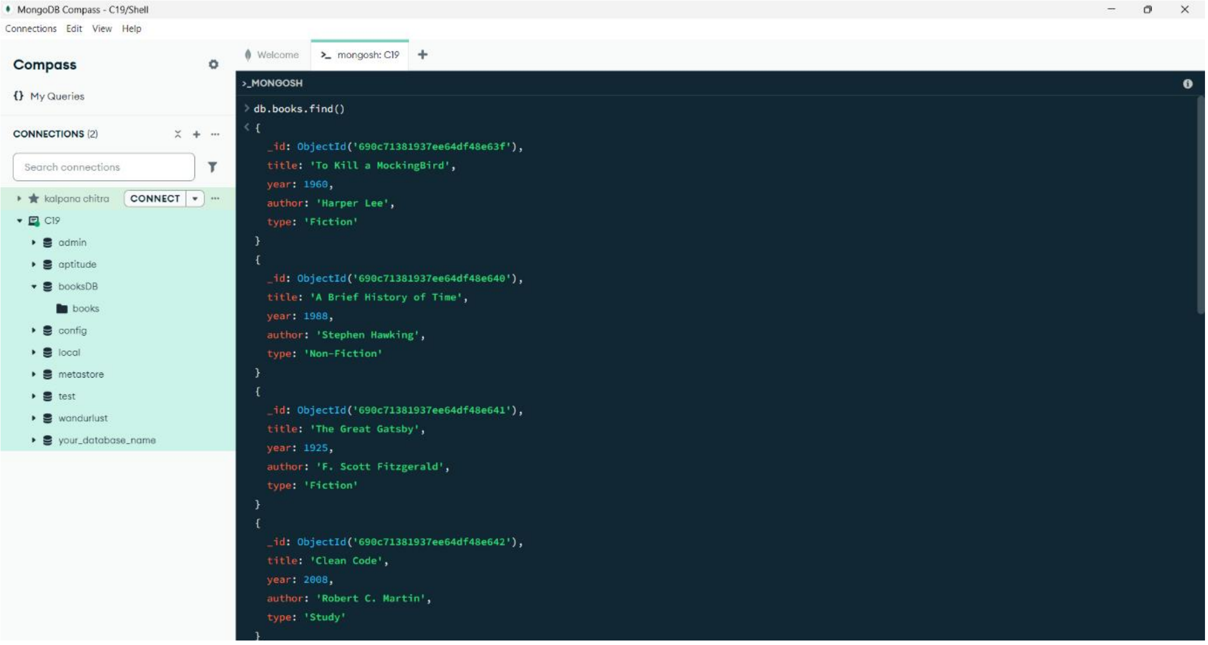
1. Inserting Books



```
> db["books"].find()
<
> db.books.insertMany(
    [
      {title: "To Kill a MockingBird", year:1960, author:"Harper Lee", type:"Fiction"},
      { title: "A Brief History of Time", year: 1988, author: "Stephen Hawking", type: "Non-Fiction" },
      { title: "The Great Gatsby", year: 1925, author: "F. Scott Fitzgerald", type: "Fiction" },
      { title: "Clean Code", year: 2008, author: "Robert C. Martin", type: "Study" },
      { title: "Educated", year: 2018, author: "Tara Westover", type: "Non-Fiction" },
      { title: "The Pragmatic Programmer", year: 1999, author: "Andrew Hunt", type: "Study" },
      { title: "1984", year: 1949, author: "George Orwell", type: "Fiction" },
      { title: "The Psychology of Money", year: 2020, author: "Morgan Housel", type: "Non-Fiction" },
      { title: "Introduction to Algorithms", year: 2009, author: "Thomas H. Cormen", type: "Study" },
      { title: "Brave New World", year: 1932, author: "Aldous Huxley", type: "Fiction" }
    ]
  )
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('690c71381937ee64df48e63f'),
      '1': ObjectId('690c71381937ee64df48e640'),
      '2': ObjectId('690c71381937ee64df48e641'),
      '3': ObjectId('690c71381937ee64df48e642'),
      '4': ObjectId('690c71381937ee64df48e643'),
      '5': ObjectId('690c71381937ee64df48e644'),
      '6': ObjectId('690c71381937ee64df48e645'),
      '7': ObjectId('690c71381937ee64df48e646'),
      '8': ObjectId('690c71381937ee64df48e647'),
      '9': ObjectId('690c71381937ee64df48e648')
```

```
db.books.insertMany(
 [
  {title: "To Kill a MockingBird", year:1960, author:"Harper Lee", type:"Fiction"},
  { title: "A Brief History of Time", year: 1988, author: "Stephen Hawking", type: "Non-Fiction" },
  { title: "The Great Gatsby", year: 1925, author: "F. Scott Fitzgerald", type: "Fiction" },
  { title: "Clean Code", year: 2008, author: "Robert C. Martin", type: "Study" },
  { title: "Educated", year: 2018, author: "Tara Westover", type: "Non-Fiction" },
  { title: "The Pragmatic Programmer", year: 1999, author: "Andrew Hunt", type: "Study" },
  { title: "1984", year: 1949, author: "George Orwell", type: "Fiction" },
  { title: "The Psychology of Money", year: 2020, author: "Morgan Housel", type: "Non-Fiction" },
  { title: "Introduction to Algorithms", year: 2009, author: "Thomas H. Cormen", type: "Study" },
  { title: "Brave New World", year: 1932, author: "Aldous Huxley", type: "Fiction" }
 ])
```

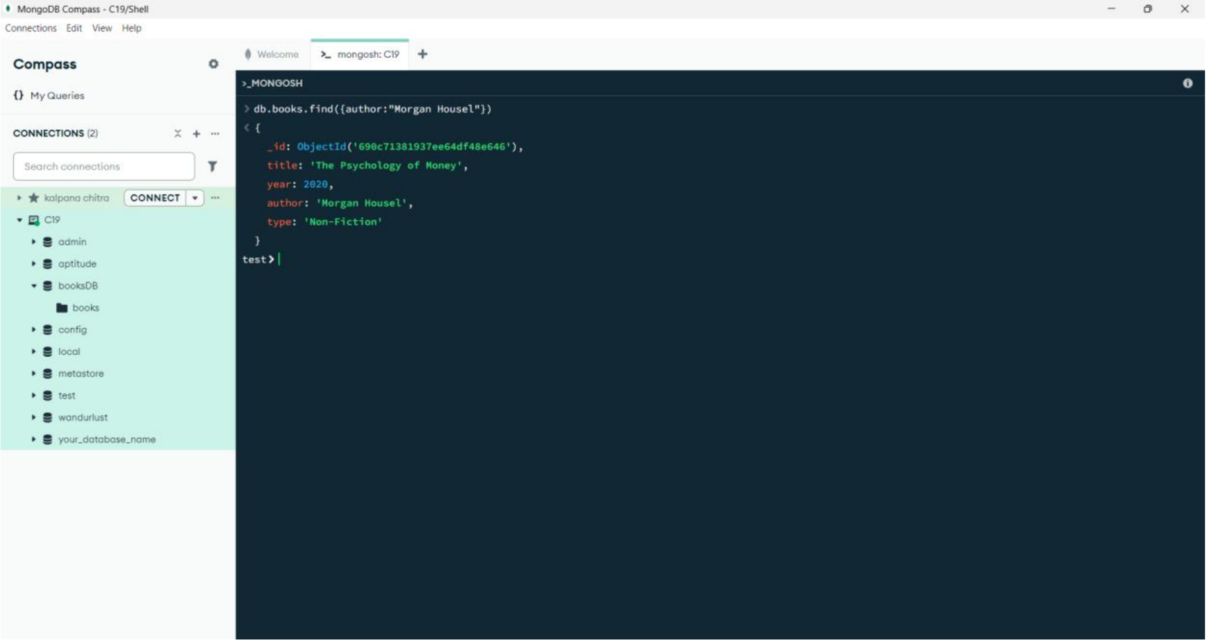
2. Show All Books



```
> db.books.find()
< {
    _id: ObjectId('690c71381937ee64df48e63f'),
    title: 'To Kill a MockingBird',
    year: 1960,
    author: 'Harper Lee',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e640'),
    title: 'A Brief History of Time',
    year: 1988,
    author: 'Stephen Hawking',
    type: 'Non-Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e641'),
    title: 'The Great Gatsby',
    year: 1925,
    author: 'F. Scott Fitzgerald',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e642'),
    title: 'Clean Code',
    year: 2008,
    author: 'Robert C. Martin',
    type: 'Study'
  }
```

db.books.find()

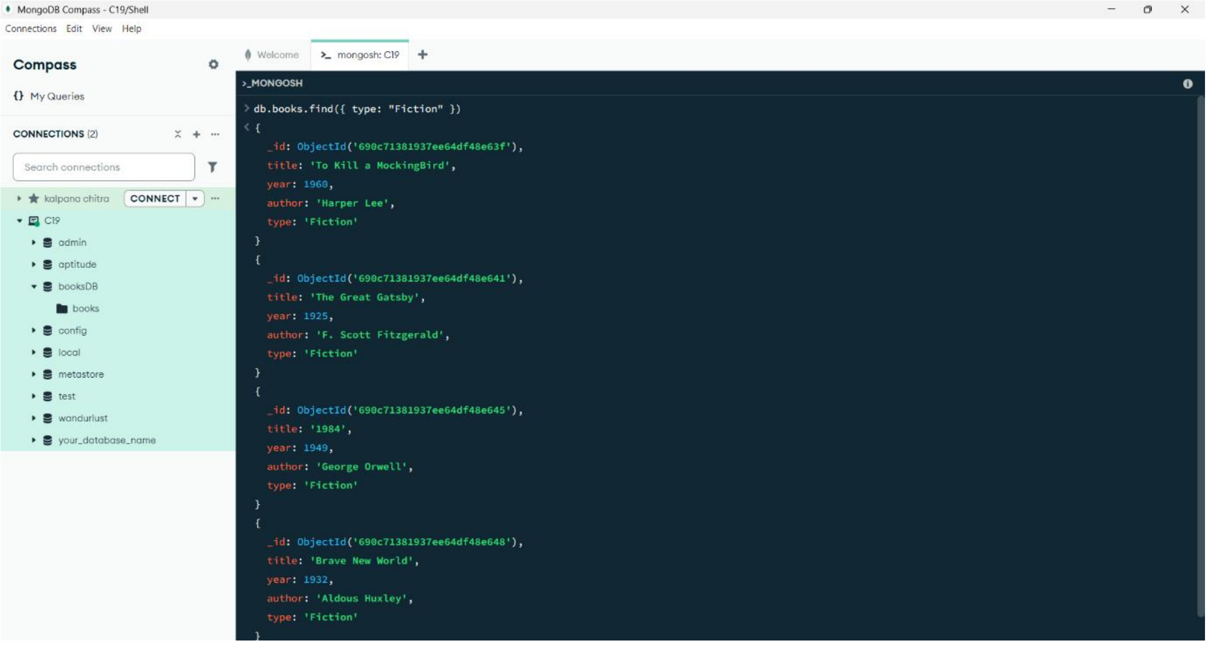
3. Find books by a specific author



```
> db.books.find({author:"Morgan Housel"})
< {
    _id: ObjectId('690c71381937ee64df48e646'),
    title: 'The Psychology of Money',
    year: 2020,
    author: 'Morgan Housel',
    type: 'Non-Fiction'
  }
test>
```

db.books.find({author:"Morgan Housel"})

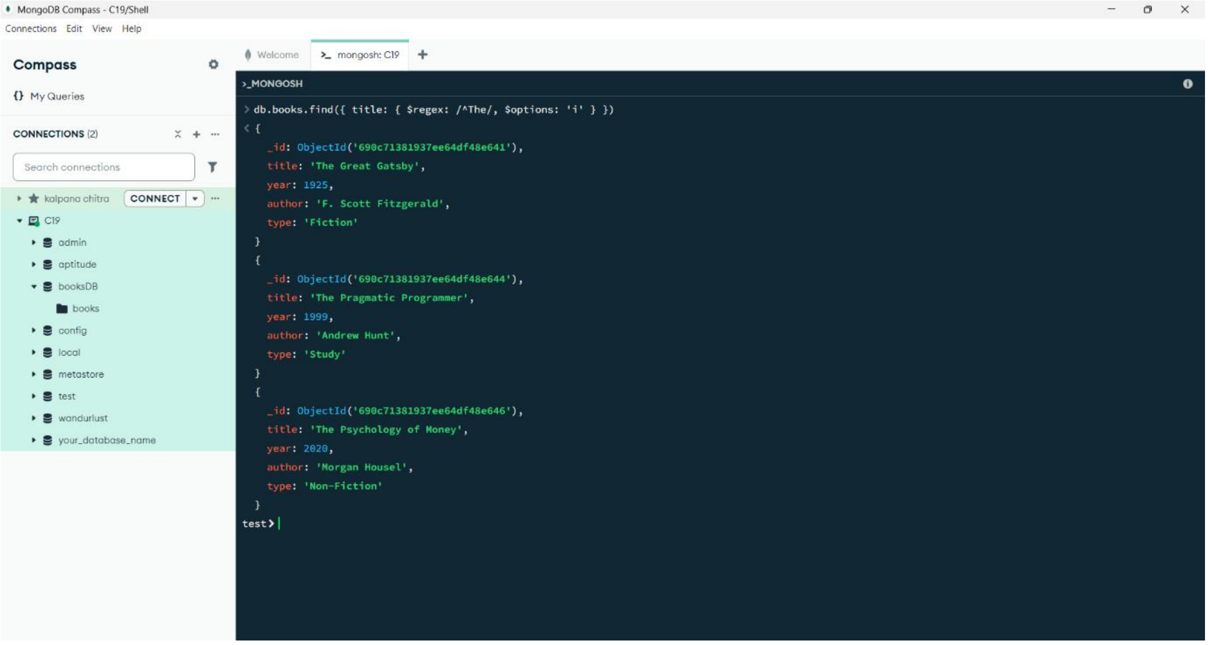
4. Find All Fiction Books



```
> db.books.find({ type: "Fiction" })
< {
    _id: ObjectId('690c71381937ee64df48e63f'),
    title: 'To Kill a MockingBird',
    year: 1960,
    author: 'Harper Lee',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e641'),
    title: 'The Great Gatsby',
    year: 1925,
    author: 'F. Scott Fitzgerald',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e645'),
    title: '1984',
    year: 1949,
    author: 'George Orwell',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e648'),
    title: 'Brave New World',
    year: 1932,
    author: 'Aldous Huxley',
    type: 'Fiction'
  }
```

db.books.find({ type: "Fiction" })

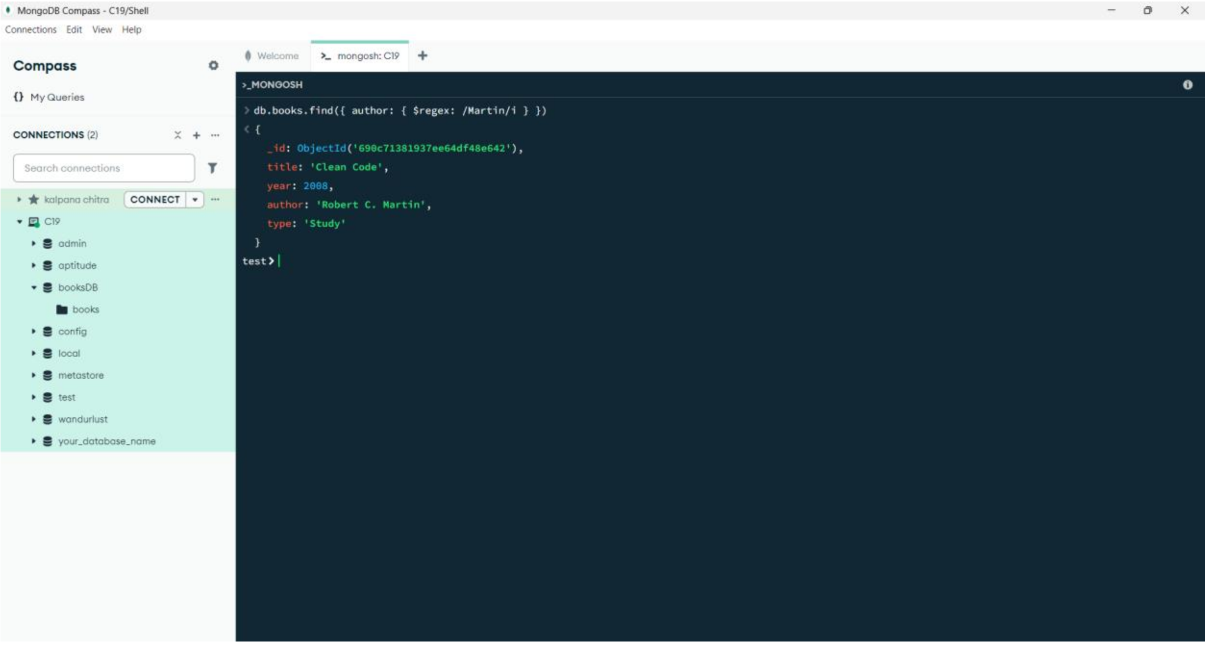
5. Find all books whose title starts with “THE”



```
> db.books.find({ title: { $regex: /^The/, $options: 'i' } })
< {
    _id: ObjectId('690c71381937ee64df48e641'),
    title: 'The Great Gatsby',
    year: 1925,
    author: 'F. Scott Fitzgerald',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e644'),
    title: 'The Pragmatic Programmer',
    year: 1999,
    author: 'Andrew Hunt',
    type: 'Study'
  }
  {
    _id: ObjectId('690c71381937ee64df48e646'),
    title: 'The Psychology of Money',
    year: 2020,
    author: 'Morgan Housel',
    type: 'Non-Fiction'
  }
test>
```

db.books.find({ title: { $regex: /^The/, $options: 'i' } })

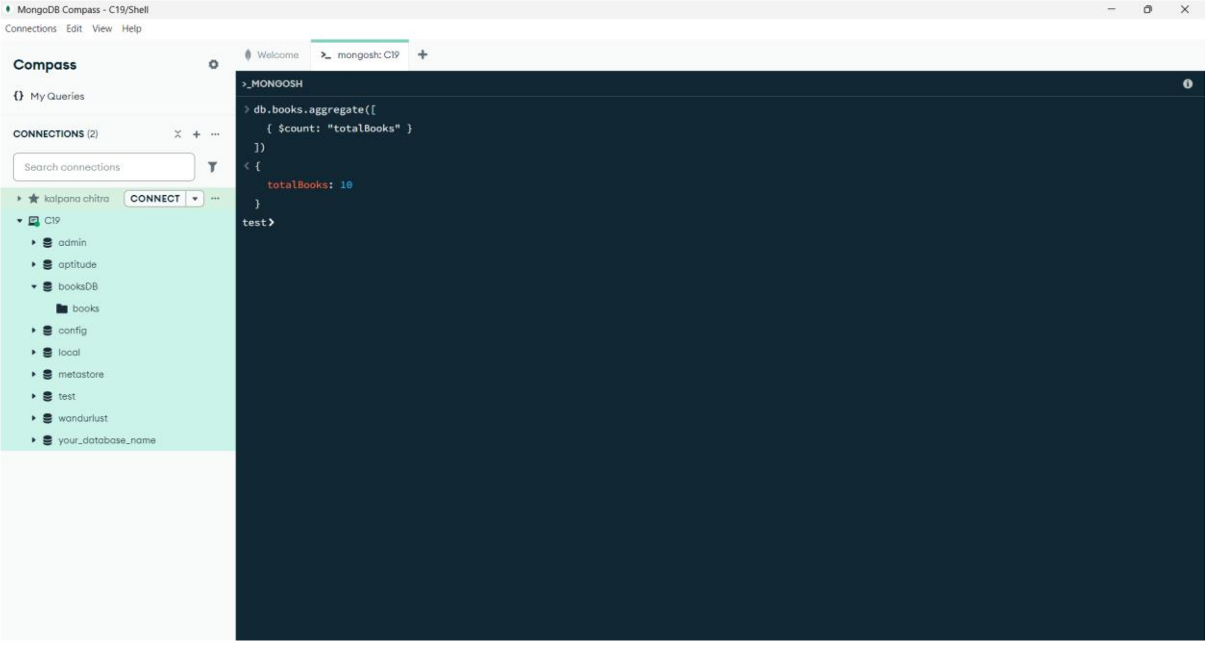
6. Find all authors whose name contains “Martin”



```
> db.books.find({ author: { $regex: /Martin/i } })
< {
    _id: ObjectId('690c71381937ee64df48e642'),
    title: 'Clean Code',
    year: 2008,
    author: 'Robert C. Martin',
    type: 'Study'
  }
test>
```

db.books.find({ author: { $regex: /Martin/i } })

7. Count Total Books



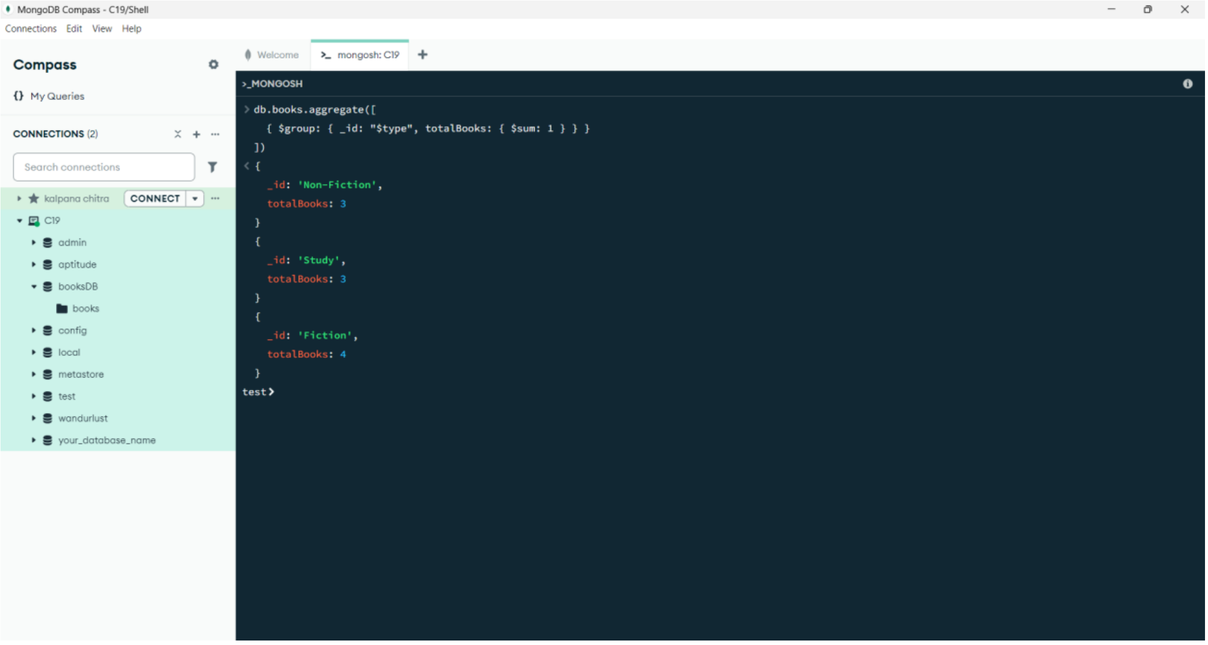
```
> db.books.aggregate([
    { $count: "totalBooks" }
  ])
< {
    totalBooks: 10
  }
test>
```

db.books.aggregate([

{ $count: "totalBooks" }

])

8. Count books by type



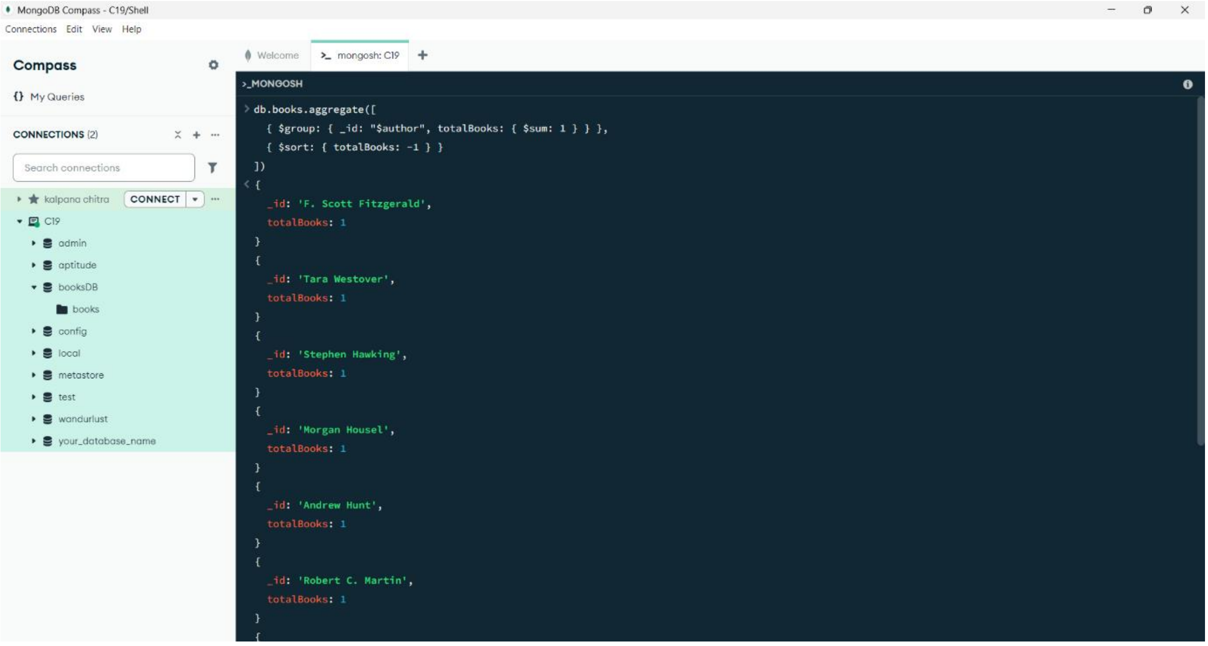
```
> db.books.aggregate([
    { $group: { _id: "$type", totalBooks: { $sum: 1 } } }
  ])
< {
    _id: 'Non-Fiction',
    totalBooks: 3
  }
  {
    _id: 'Study',
    totalBooks: 3
  }
  {
    _id: 'Fiction',
    totalBooks: 4
  }
test>
```

db.books.aggregate([

{ $group: { _id: "$type", totalBooks: { $sum: 1 } } }

])

9. Count books per author



```
> db.books.aggregate([
    { $group: { _id: "$author", totalBooks: { $sum: 1 } } },
    { $sort: { totalBooks: -1 } }
  ])
< {
    _id: 'F. Scott Fitzgerald',
    totalBooks: 1
  }
  {
    _id: 'Tara Westover',
    totalBooks: 1
  }
  {
    _id: 'Stephen Hawking',
    totalBooks: 1
  }
  {
    _id: 'Morgan Housel',
    totalBooks: 1
  }
  {
    _id: 'Andrew Hunt',
    totalBooks: 1
  }
  {
    _id: 'Robert C. Martin',
    totalBooks: 1
  }
  {
```

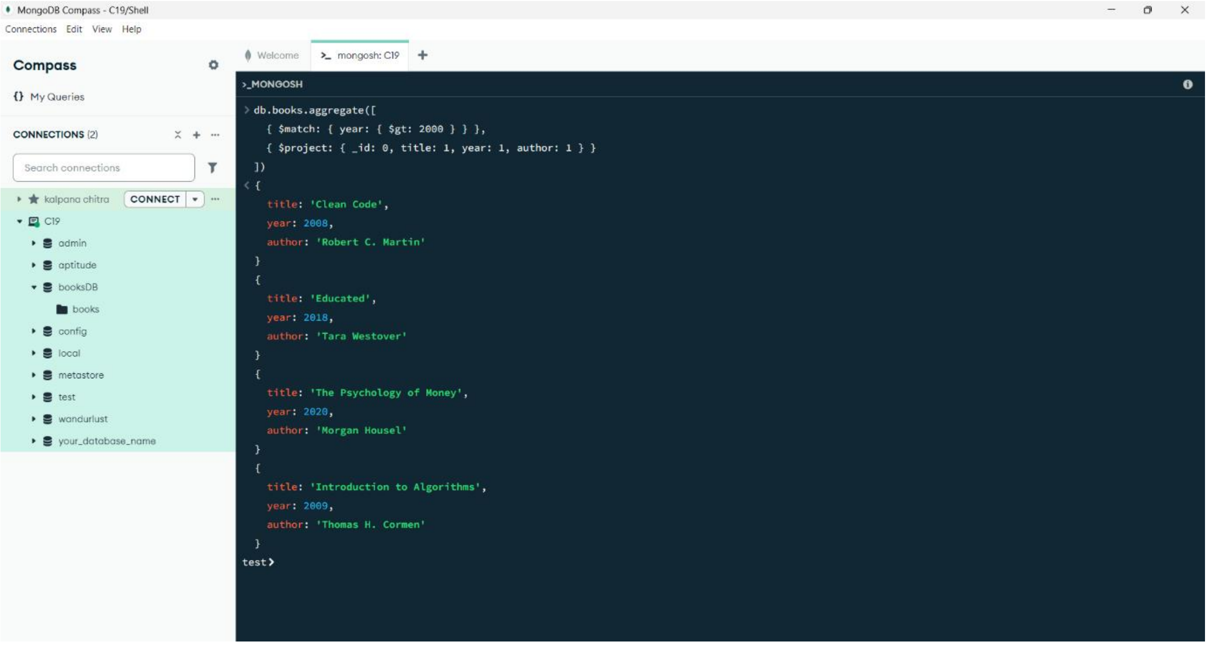
db.books.aggregate([

{ $group: { _id: "$author", totalBooks: { $sum: 1 } } },

{ $sort: { totalBooks: -1 } }

])

10. Books published after year 2000



```
> db.books.aggregate([
    { $match: { year: { $gt: 2000 } } },
    { $project: { _id: 0, title: 1, year: 1, author: 1 } }
  ])
< {
    title: 'Clean Code',
    year: 2008,
    author: 'Robert C. Martin'
  }
  {
    title: 'Educated',
    year: 2018,
    author: 'Tara Westover'
  }
  {
    title: 'The Psychology of Money',
    year: 2020,
    author: 'Morgan Housel'
  }
  {
    title: 'Introduction to Algorithms',
    year: 2009,
    author: 'Thomas H. Cormen'
  }
test>
```

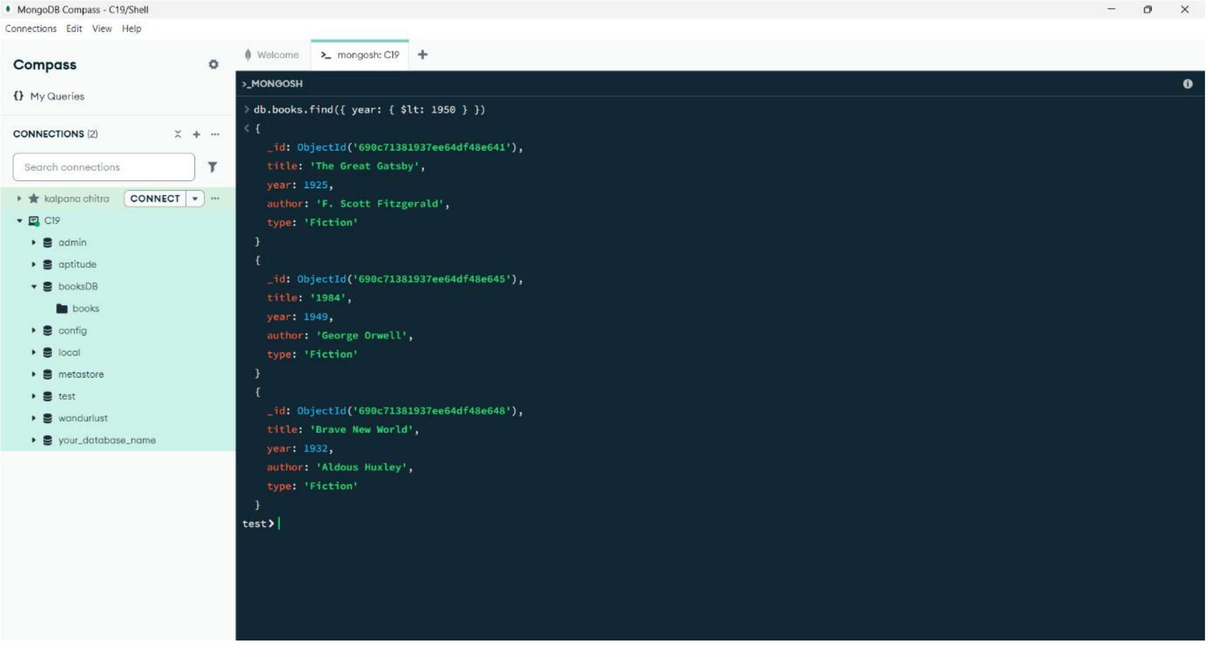
db.books.aggregate([

{ $match: { year: { $gt: 2000 } } },

{ $project: { _id: 0, title: 1, year: 1, author: 1 } }

])

11. Find books published before 1950

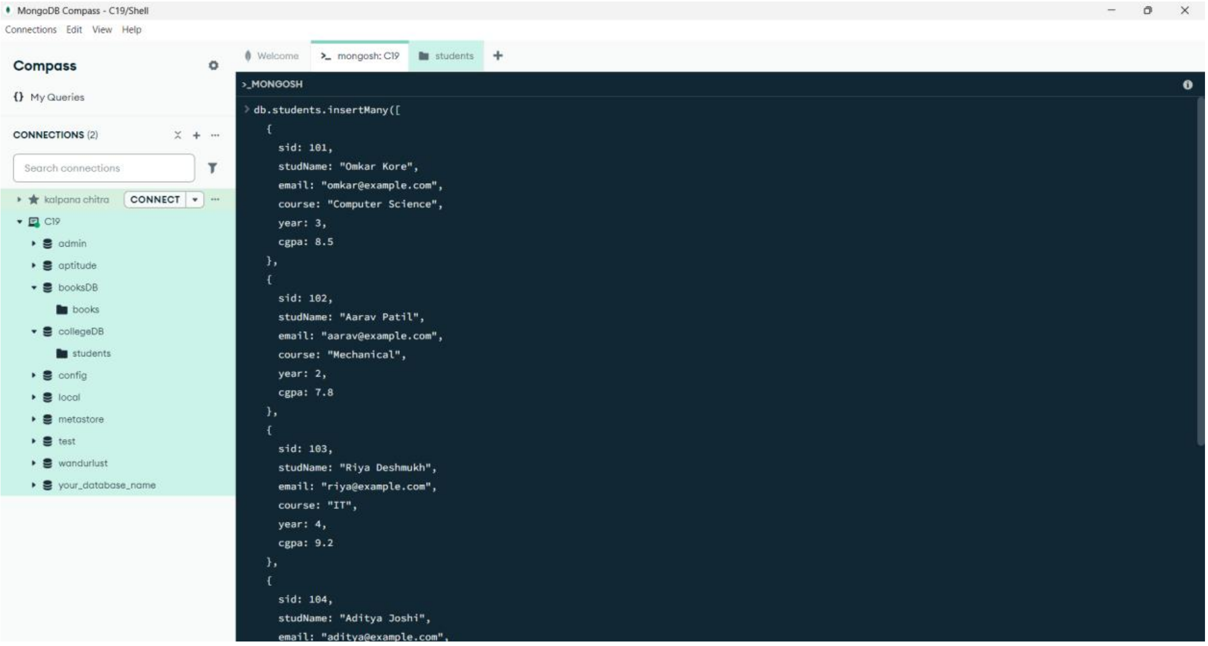


```
> db.books.find({ year: { $lt: 1950 } })
< {
    _id: ObjectId('690c71381937ee64df48e641'),
    title: 'The Great Gatsby',
    year: 1925,
    author: 'F. Scott Fitzgerald',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e645'),
    title: '1984',
    year: 1949,
    author: 'George Orwell',
    type: 'Fiction'
  }
  {
    _id: ObjectId('690c71381937ee64df48e648'),
    title: 'Brave New World',
    year: 1932,
    author: 'Aldous Huxley',
    type: 'Fiction'
  }
test>
```

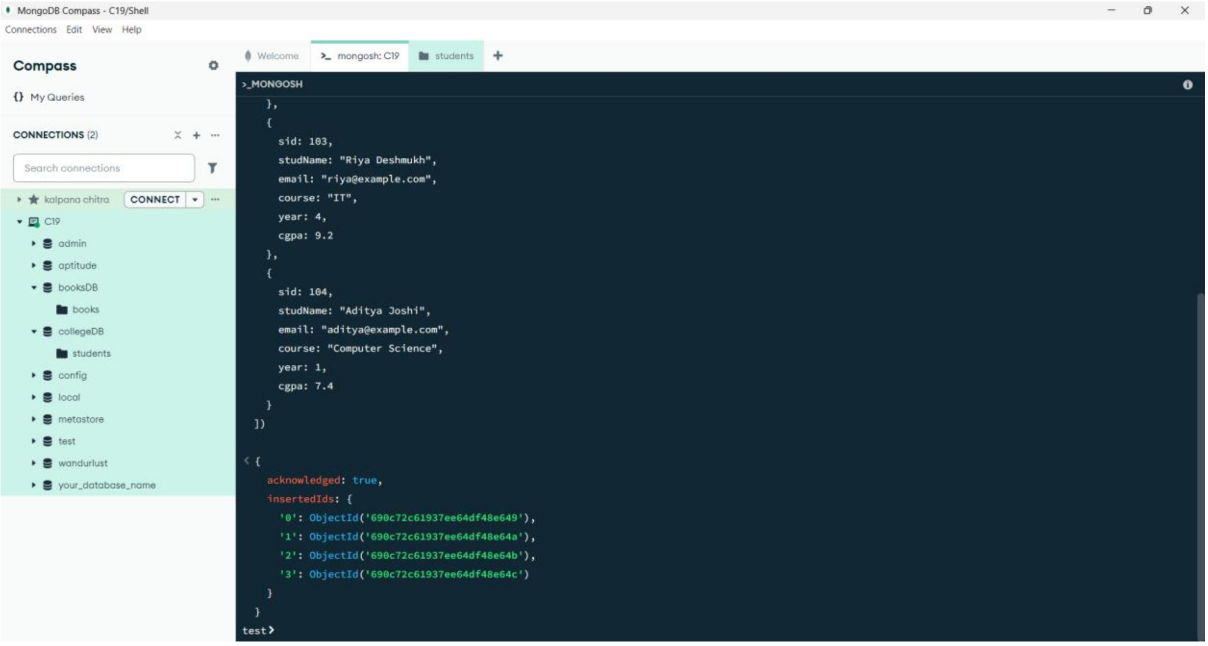
db.books.find({ year: { $lt: 1950 } })

PART 2:- Student Database

1. INSERTING STUDENTS



```
> db.students.insertMany([
    {
      sid: 101,
      studName: "Omkar Kore",
      email: "omkar@example.com",
      course: "Computer Science",
      year: 3,
      cgpa: 8.5
    },
    {
      sid: 102,
      studName: "Aarav Patil",
      email: "aarav@example.com",
      course: "Mechanical",
      year: 2,
      cgpa: 7.8
    },
    {
      sid: 103,
      studName: "Riya Deshmukh",
      email: "riya@example.com",
      course: "IT",
      year: 4,
      cgpa: 9.2
    },
    {
      sid: 104,
      studName: "Aditya Joshi",
      email: "aditya@example.com",
```

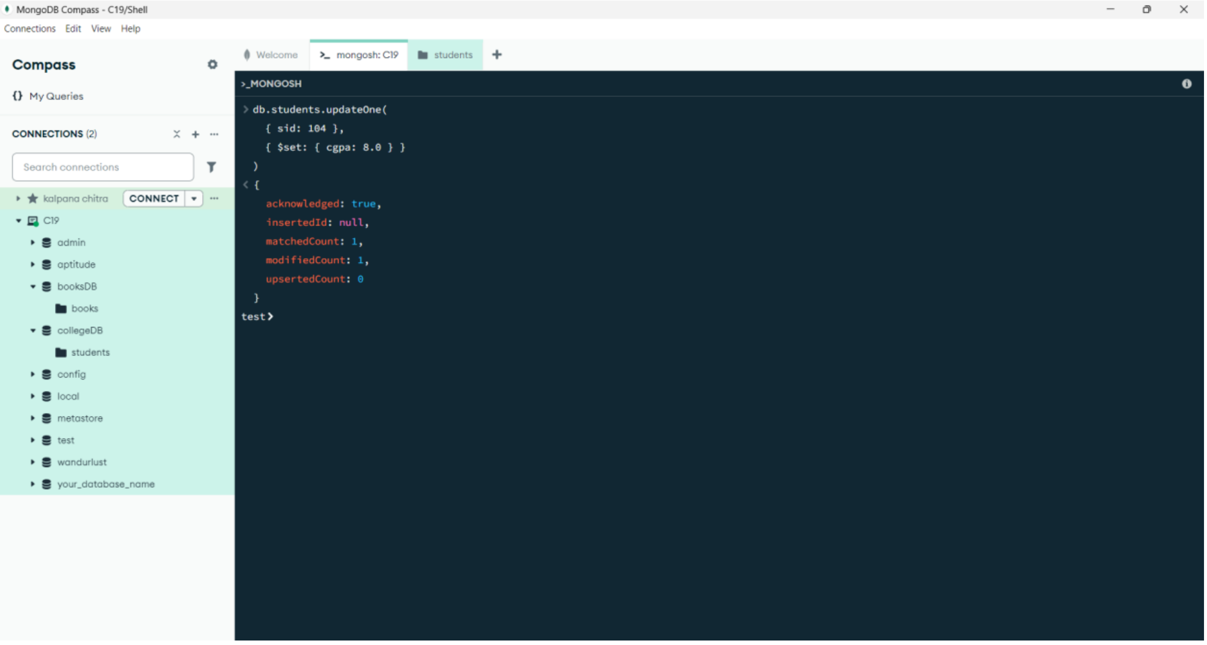


```
    },
    {
      sid: 103,
      studName: "Riya Deshmukh",
      email: "riya@example.com",
      course: "IT",
      year: 4,
      cgpa: 9.2
    },
    {
      sid: 104,
      studName: "Aditya Joshi",
      email: "aditya@example.com",
      course: "Computer Science",
      year: 1,
      cgpa: 7.4
    }
  ])

< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('690c72c61937ee64df48e649'),
      '1': ObjectId('690c72c61937ee64df48e64a'),
      '2': ObjectId('690c72c61937ee64df48e64b'),
      '3': ObjectId('690c72c61937ee64df48e64c')
    }
  }
test>
```

```
db.students.insertMany([
  {
    sid: 101,
    studName: "Omkar Kore",
    email: "omkar@example.com",
    course: "Computer Science",
    year: 3,
    cgpa: 8.5
  },
  {
    sid: 102,
    studName: "Aarav Patil",
    email: "aarav@example.com",
    course: "Mechanical",
    year: 2,
    cgpa: 7.8
  },
  {
    sid: 103,
    studName: "Riya Deshmukh",
    email: "riya@example.com",
    course: "IT",
    year: 4,
    cgpa: 9.2
  },
  {
    sid: 104,
    studName: "Aditya Joshi",
    email: "aditya@example.com",
    course: "Computer Science",
    year: 1,
    cgpa: 7.4
  }
])
```

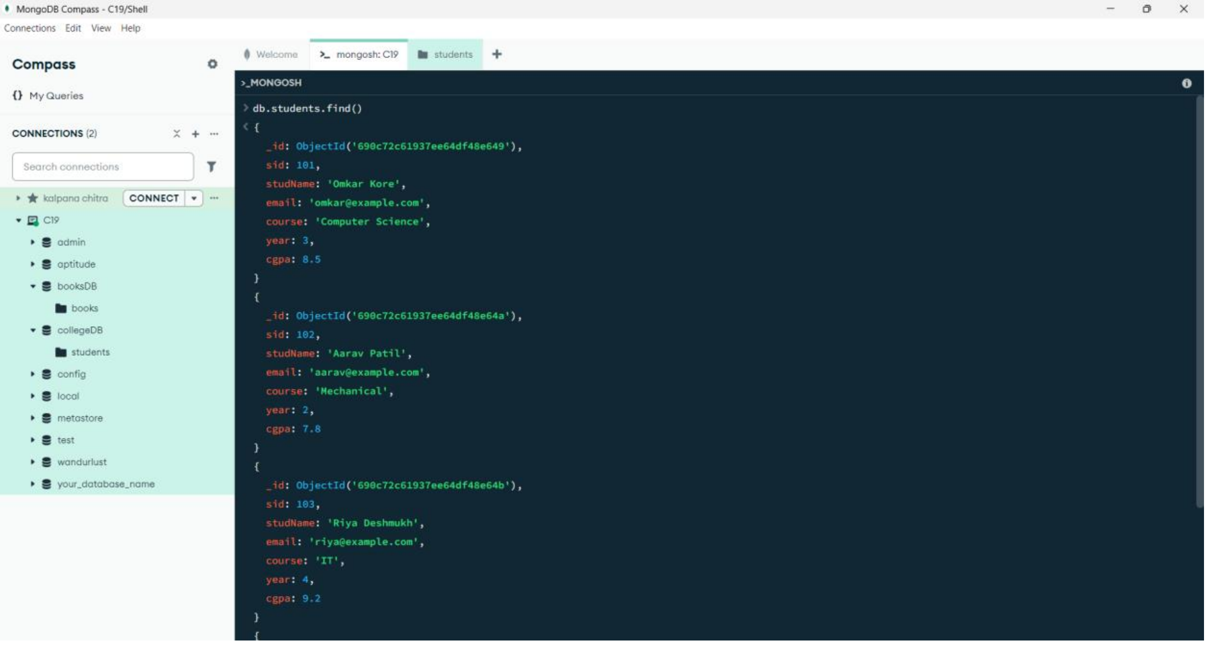
2. Updating student cgpa



```
> db.students.updateOne(
    { sid: 104 },
    { $set: { cgpa: 8.0 } }
  )
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
test>
```

```
db.students.updateOne(
  { sid: 104 },
  { $set: { cgpa: 8.0 } }
)
```

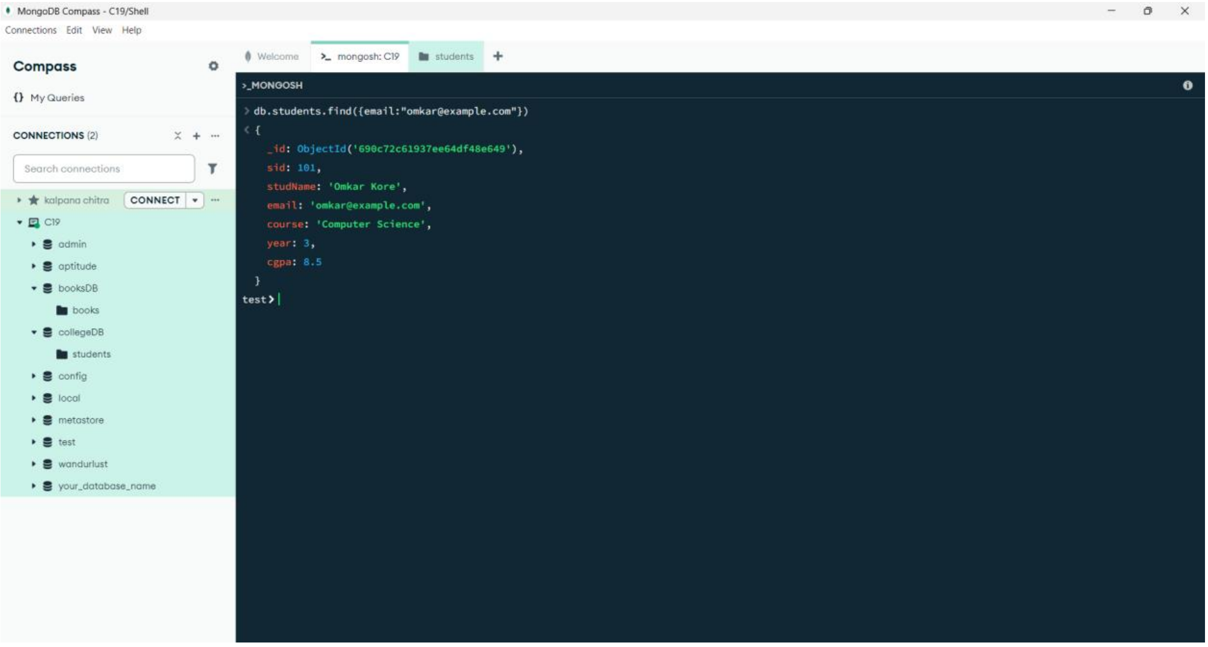
3. Get all students



```
> db.students.find()
< {
    _id: ObjectId('690c72c61937ee64df48e649'),
    sid: 101,
    studName: 'Omkar Kore',
    email: 'omkar@example.com',
    course: 'Computer Science',
    year: 3,
    cgpa: 8.5
  }
  {
    _id: ObjectId('690c72c61937ee64df48e64a'),
    sid: 102,
    studName: 'Aarav Patil',
    email: 'aarav@example.com',
    course: 'Mechanical',
    year: 2,
    cgpa: 7.8
  }
  {
    _id: ObjectId('690c72c61937ee64df48e64b'),
    sid: 103,
    studName: 'Riya Deshmukh',
    email: 'riya@example.com',
    course: 'IT',
    year: 4,
    cgpa: 9.2
  }
  {
```

db.students.find()

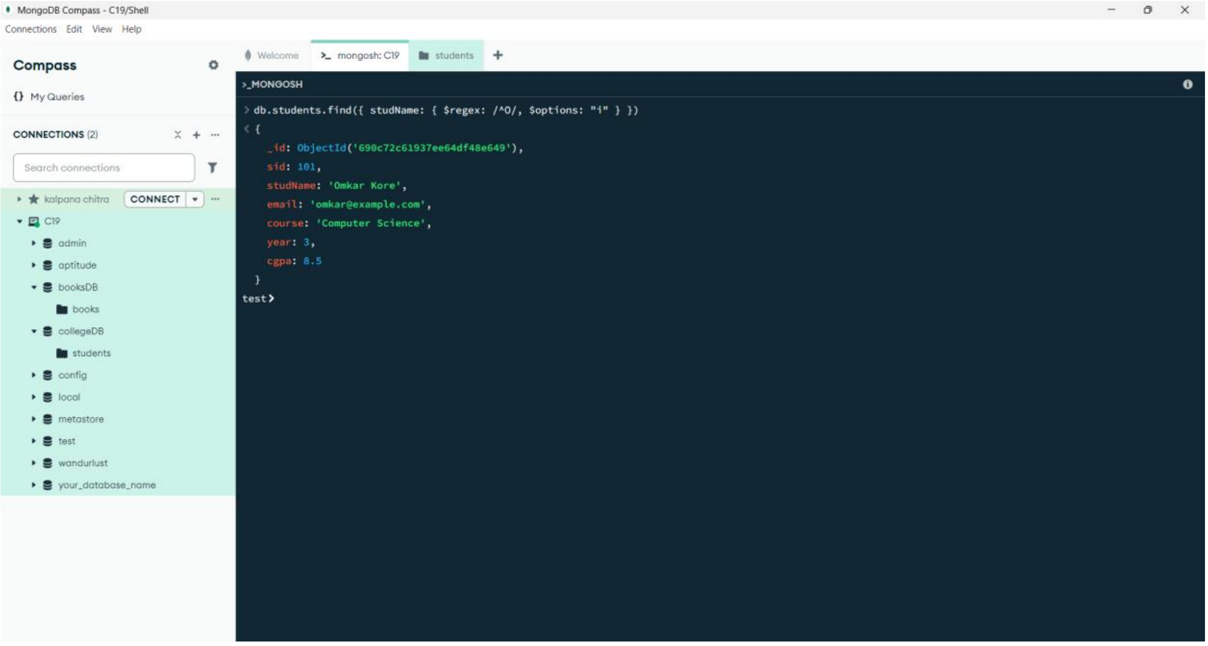
4. Get student with email omkar@example.com



```
> db.students.find({email:"omkar@example.com"})
< {
    _id: ObjectId('690c72c61937ee64df48e649'),
    sid: 101,
    studName: 'Omkar Kore',
    email: 'omkar@example.com',
    course: 'Computer Science',
    year: 3,
    cgpa: 8.5
  }
test>
```

db.students.find({email:"omkar@example.com"})

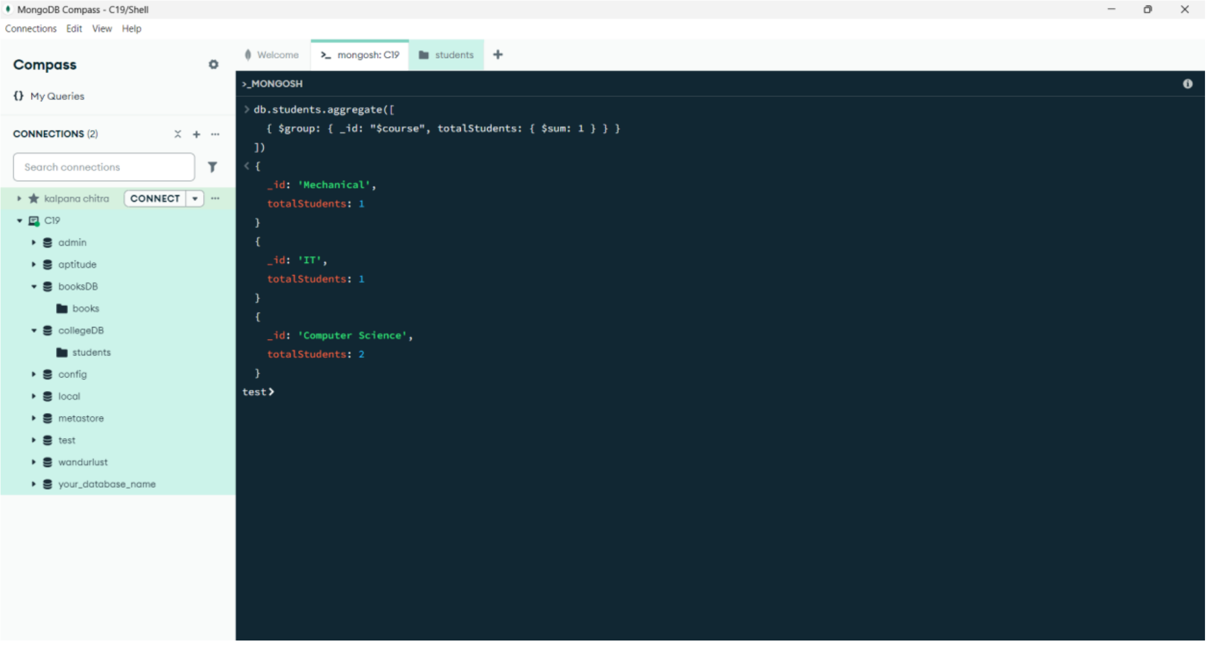
5. Find names starting with O



```
> db.students.find({ studName: { $regex: /^O/, $options: "i" } })
< {
    _id: ObjectId('690c72c61937ee64df48e649'),
    sid: 101,
    studName: 'Omkar Kore',
    email: 'omkar@example.com',
    course: 'Computer Science',
    year: 3,
    cgpa: 8.5
  }
test>
```

db.students.find({ studName: { $regex: /^O/, $options: "i" } })

6. Count students in each course



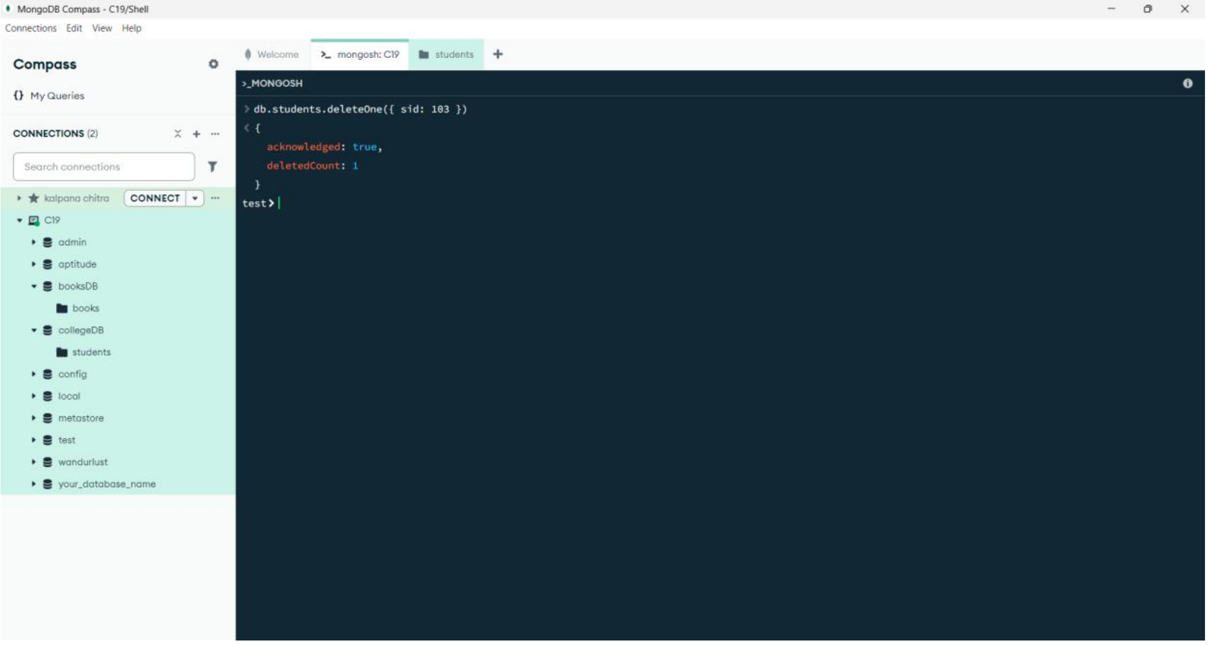
```
> db.students.aggregate([
    { $group: { _id: "$course", totalStudents: { $sum: 1 } } }
  ])
< {
    _id: 'Mechanical',
    totalStudents: 1
  }
  {
    _id: 'IT',
    totalStudents: 1
  }
  {
    _id: 'Computer Science',
    totalStudents: 2
  }
test>
```

db.students.aggregate([

{ $group: { _id: "$course", totalStudents: { $sum: 1 } } }

])

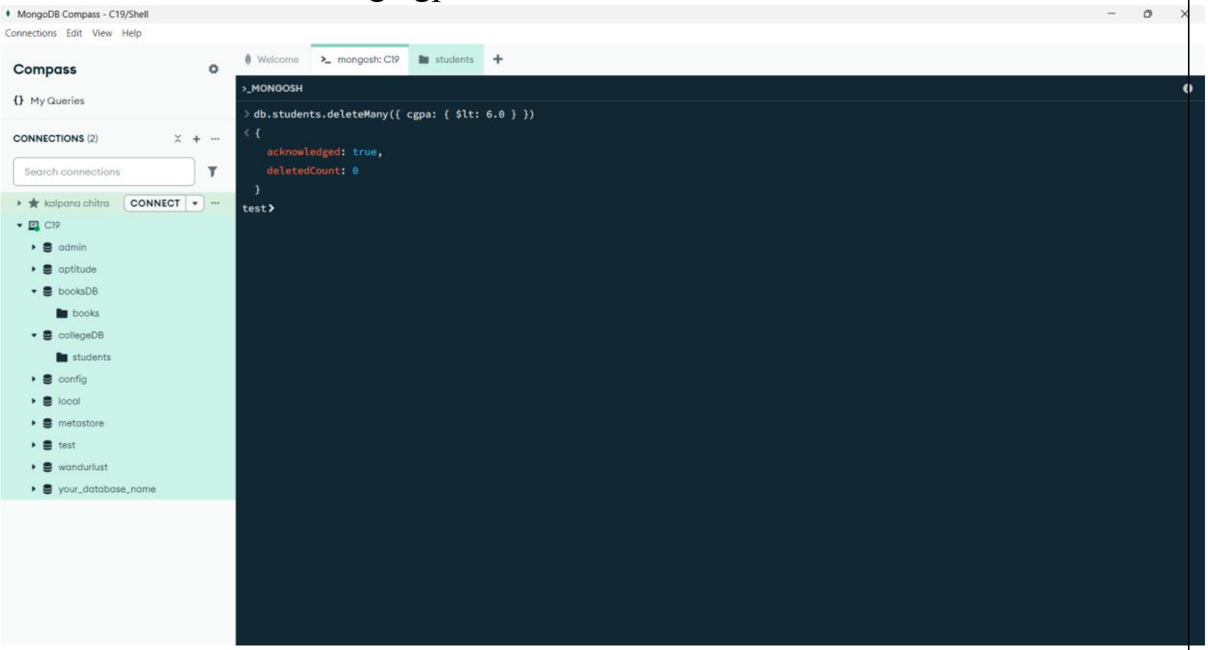
7. Delete a student



```
> db.students.deleteOne({ sid: 103 })
< {
    acknowledged: true,
    deletedCount: 1
  }
test>
```

db.students.deleteOne({ sid: 102 })

8. Delete all students having cgpa below 6



```
> db.students.deleteMany({ cgpa: { $lt: 6.0 } })
< {
    acknowledged: true,
    deletedCount: 0
  }
test>
```

db.students.deleteMany({ cgpa: { $lt: 6.0 } })