Day 8 Coding Assignment — Indexing, Aggregation, and MongoDB Atlas

Scenario Background

BookVerse is now scaling, and you are tasked with optimizing database performance and enabling analytics using MongoDB's advanced features. You will implement indexes, run aggregation pipelines, and connect your local database to MongoDB Atlas.

User Story 1 —

Indexing and Query Optimization As a developer, I need to optimize frequently used queries to improve database performance.

Tasks:

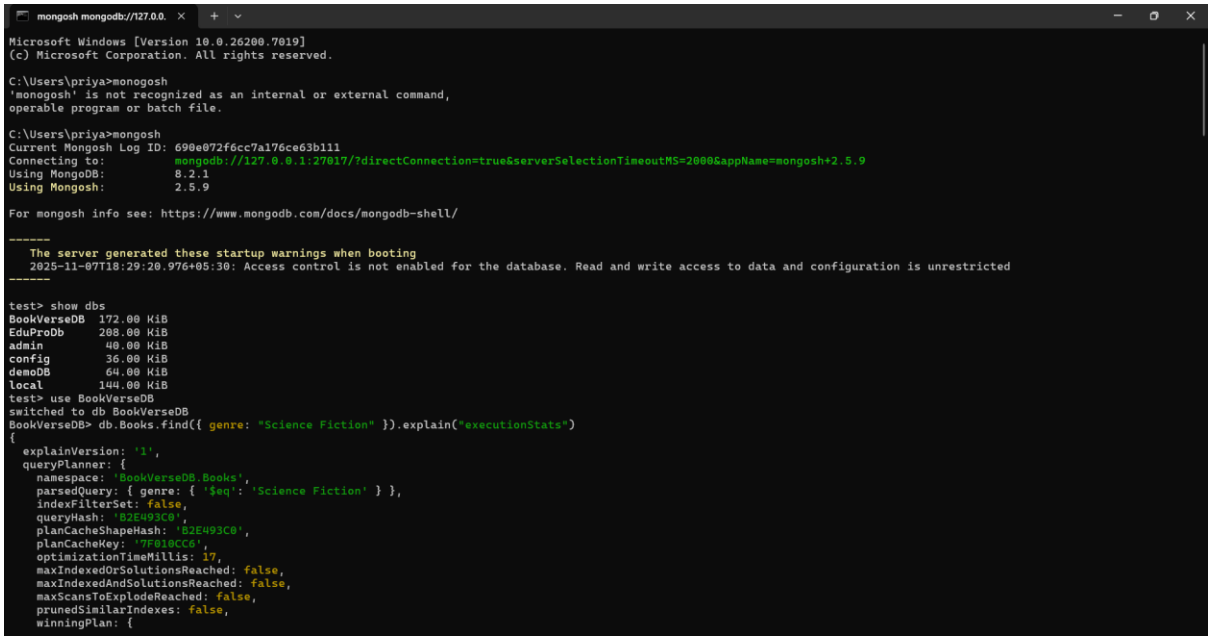
1. Identify frequently queried fields (e.g., genre, authorId, ratings.score).

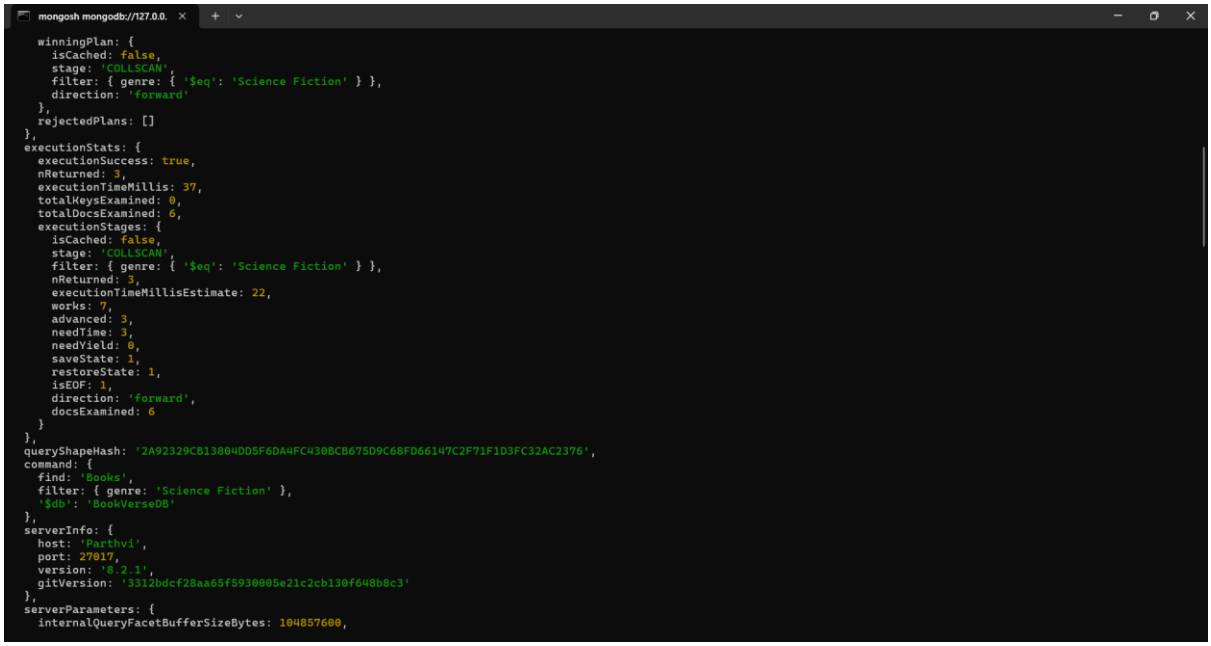
2. Create indexes on these fields using the createIndex() method.

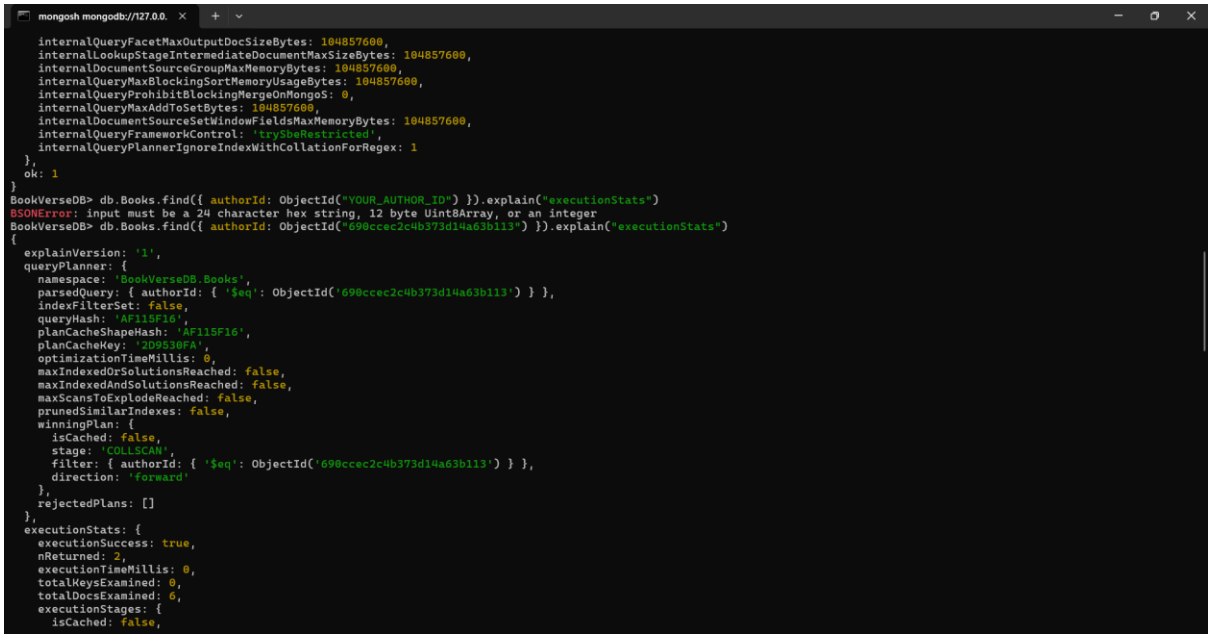
3. Compare query performance using the explain("executionStats") command before and after creating indexes.

4. Drop an unnecessary index using dropIndex() and note its impact.

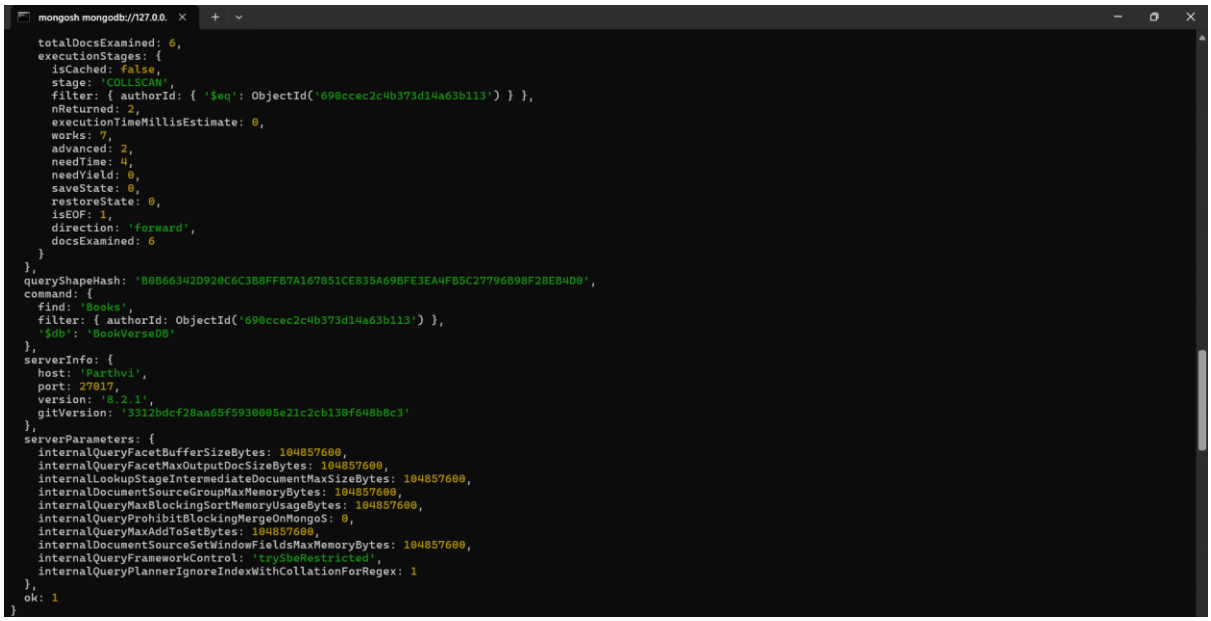
Concepts Covered: Indexing, query optimization, performance analysis


mongosh mongodb://127.0.0.
Microsoft Windows [Version 10.0.26200.7019]
(c) Microsoft Corporation. All rights reserved.
C:\Users\priya>monogosh
'monogosh' is not recognized as an internal or external command,
operable program or batch file.
C:\Users\priya>mongosh
Current Mongosh Log ID: 690e072f6cc7a176ce63b111
Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.9
Using MongoDB: 8.2.1
Using Mongosh: 2.5.9
For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
The server generated these startup warnings when booting
2025-11-07T18:29:20.976+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
test> show dbs
BookVerseDB 172.00 KiB
EduProDb 208.00 KiB
admin 40.00 KiB
config 36.00 KiB
demoDB 64.00 KiB
local 144.00 KiB
test> use BookVerseDB
switched to db BookVerseDB
BookVerseDB> db.Books.find({ genre: "Science Fiction" }).explain("executionStats")
{
explainVersion: '1',
queryPlanner: {
namespace: 'BookVerseDB.Books',
parsedQuery: { genre: { '$eq': 'Science Fiction' } },
indexFilterSet: false,
queryHash: 'B2E493C0',
planCacheShapeHash: 'B2E493C0',
planCacheKey: '7F010CC6',
optimizationTimeMillis: 17,
maxIndexedOrSolutionsReached: false,
maxIndexedAndSolutionsReached: false,
maxScansToExplodeReached: false,
prunedSimilarIndexes: false,
winningPlan: {

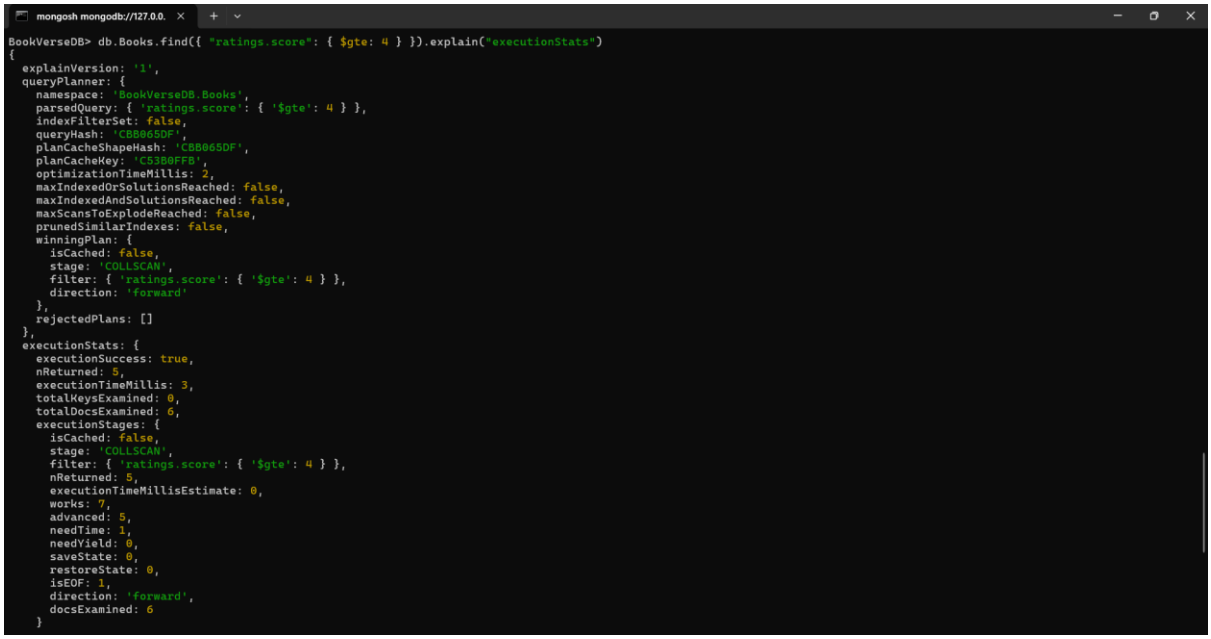

mongosh mongodb://127.0.0.
winningPlan: {
isCached: false,
stage: 'COLLSCAN',
filter: { genre: { '$eq': 'Science Fiction' } },
direction: 'forward'
},
rejectedPlans: []
},
executionStats: {
executionSuccess: true,
nReturned: 3,
executionTimeMillis: 37,
totalKeysExamined: 0,
totalDocsExamined: 6,
executionStages: {
isCached: false,
stage: 'COLLSCAN',
filter: { genre: { '$eq': 'Science Fiction' } },
nReturned: 3,
executionTimeMillisEstimate: 22,
works: 7,
advanced: 3,
needTime: 3,
needYield: 0,
saveState: 1,
restoreState: 1,
isEOF: 1,
direction: 'forward',
docsExamined: 6
}
},
queryShapeHash: '2A92329CB13804DD5F6DA4FC430BCB675D9C68FD66147C2F71F1D3FC32AC2376',
command: {
find: 'Books',
filter: { genre: 'Science Fiction' },
'$db': 'BookVerseDB'
},
serverInfo: {
host: 'Parthvi',
port: 27017,
version: '8.2.1',
gitVersion: '3312bdcf28aa65f5930005e21c2cb130f648b8c3'
},
serverParameters: {
internalQueryFacetBufferSizeBytes: 104857600,



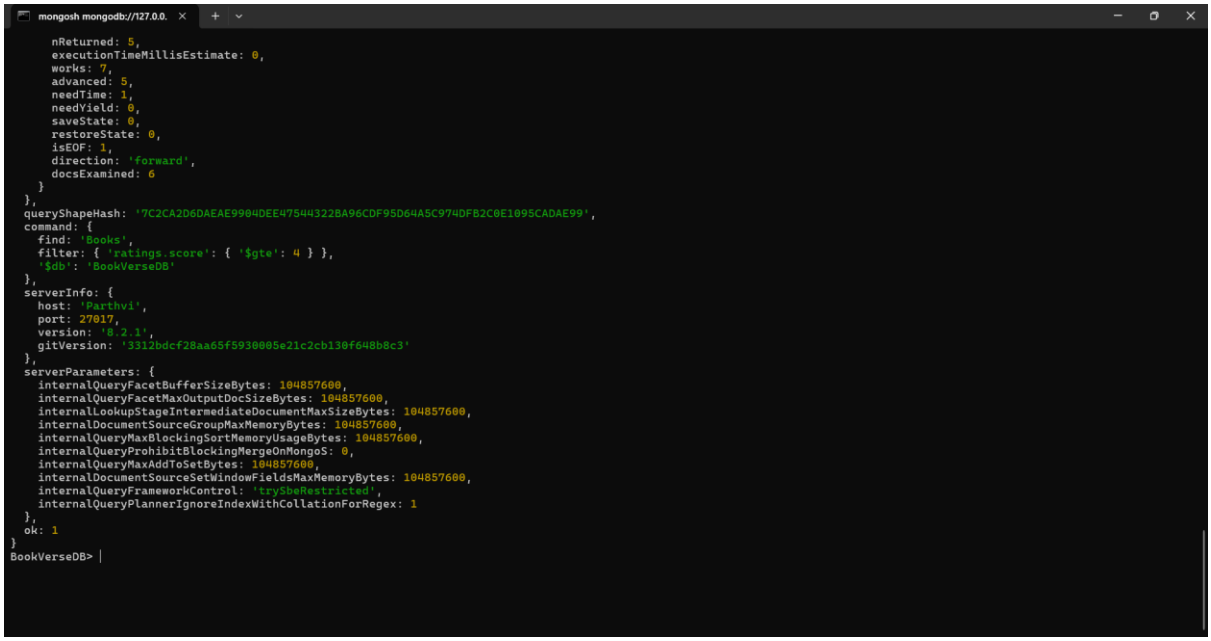
```
mongosh mongodb://127.0.0.

    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    internalQueryMaxAddToSetBytes: 104857600,
    internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
    internalQueryFrameworkControl: 'trySbeRestricted',
    internalQueryPlannerIgnoreIndexWithCollationForRegex: 1
  },
  ok: 1
}
BookVerseDB> db.Books.find({ authorId: ObjectId("YOUR_AUTHOR_ID") }).explain("executionStats")
BSONError: input must be a 24 character hex string, 12 byte Uint8Array, or an integer
BookVerseDB> db.Books.find({ authorId: ObjectId("690ccec2c4b373d14a63b113") }).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'BookVerseDB.Books',
    parsedQuery: { authorId: { '$eq': ObjectId('690ccec2c4b373d14a63b113') } },
    indexFilterSet: false,
    queryHash: 'AF115F16',
    planCacheShapeHash: 'AF115F16',
    planCacheKey: '2D9530FA',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { authorId: { '$eq': ObjectId('690ccec2c4b373d14a63b113') } },
      direction: 'forward'
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 2,
    executionTimeMillis: 0,
    totalKeysExamined: 0,
    totalDocsExamined: 6,
    executionStages: {
      isCached: false,
```



```
mongosh mongodb://127.0.0.

    totalDocsExamined: 6,
    executionStages: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { authorId: { '$eq': ObjectId('690ccec2c4b373d14a63b113') } },
      nReturned: 2,
      executionTimeMillisEstimate: 0,
      works: 7,
      advanced: 2,
      needTime: 4,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      direction: 'forward',
      docsExamined: 6
    }
  },
  queryShapeHash: 'B0B66342D920C6C3B8FFB7A167851CE835A69BFE3EA4FB5C27796B98F28EB4D0',
  command: {
    find: 'Books',
    filter: { authorId: ObjectId('690ccec2c4b373d14a63b113') },
    '$db': 'BookVerseDB'
  },
  serverInfo: {
    host: 'Parthvi',
    port: 27017,
    version: '8.2.1',
    gitVersion: '3312bdcf28aa65f5930005e21c2cb130f648b8c3'
  },
  serverParameters: {
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    internalQueryMaxAddToSetBytes: 104857600,
    internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
    internalQueryFrameworkControl: 'trySbeRestricted',
    internalQueryPlannerIgnoreIndexWithCollationForRegex: 1
  },
  ok: 1
}
```

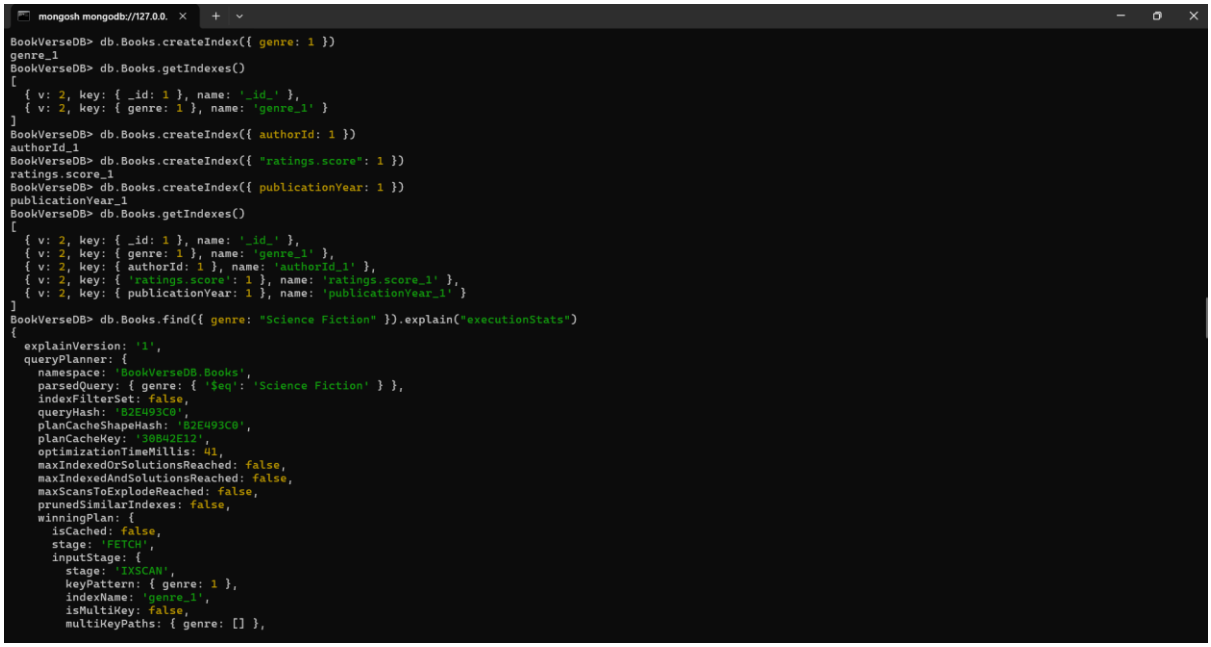


```
mongosh mongodb://127.0.0.

BookVerseDB> db.Books.find({ "ratings.score": { $gte: 4 } }).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'BookVerseDB.Books',
    parsedQuery: { 'ratings.score': { '$gte': 4 } },
    indexFilterSet: false,
    queryHash: 'CBB065DF',
    planCacheShapeHash: 'CBB065DF',
    planCacheKey: 'C53B0FFB',
    optimizationTimeMillis: 2,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { 'ratings.score': { '$gte': 4 } },
      direction: 'forward'
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 5,
    executionTimeMillis: 3,
    totalKeysExamined: 0,
    totalDocsExamined: 6,
    executionStages: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { 'ratings.score': { '$gte': 4 } },
      nReturned: 5,
      executionTimeMillisEstimate: 0,
      works: 7,
      advanced: 5,
      needTime: 1,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      direction: 'forward',
      docsExamined: 6
    }
```

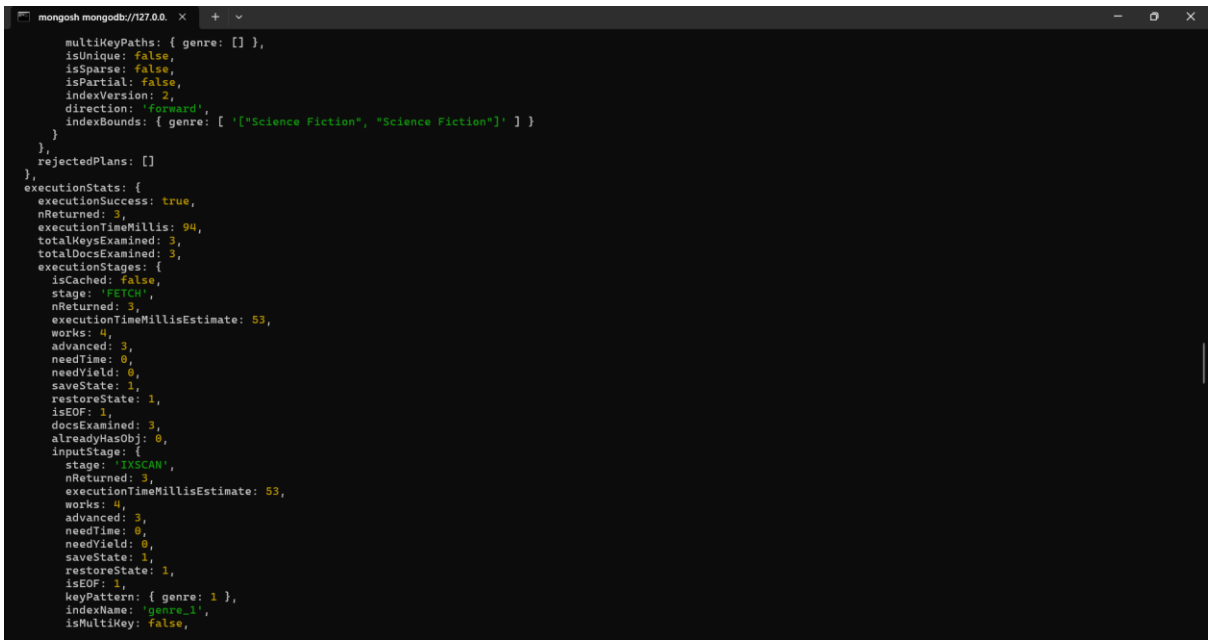


```
mongosh mongodb://127.0.0.

      nReturned: 5,
      executionTimeMillisEstimate: 0,
      works: 7,
      advanced: 5,
      needTime: 1,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      direction: 'forward',
      docsExamined: 6
    }
  },
  queryShapeHash: '7C2CA2D6DAEAE9904DEE47544322BA96CDF95D64A5C974DFB2C0E1095CADAE99',
  command: {
    find: 'Books',
    filter: { 'ratings.score': { '$gte': 4 } },
    '$db': 'BookVerseDB'
  },
  serverInfo: {
    host: 'Parthvi',
    port: 27017,
    version: '8.2.1',
    gitVersion: '3312bdcf28aa65f5930005e21c2cb130f648b8c3'
  },
  serverParameters: {
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    internalQueryMaxAddToSetBytes: 104857600,
    internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
    internalQueryFrameworkControl: 'trySbeRestricted',
    internalQueryPlannerIgnoreIndexWithCollationForRegex: 1
  },
  ok: 1
}
BookVerseDB>
```

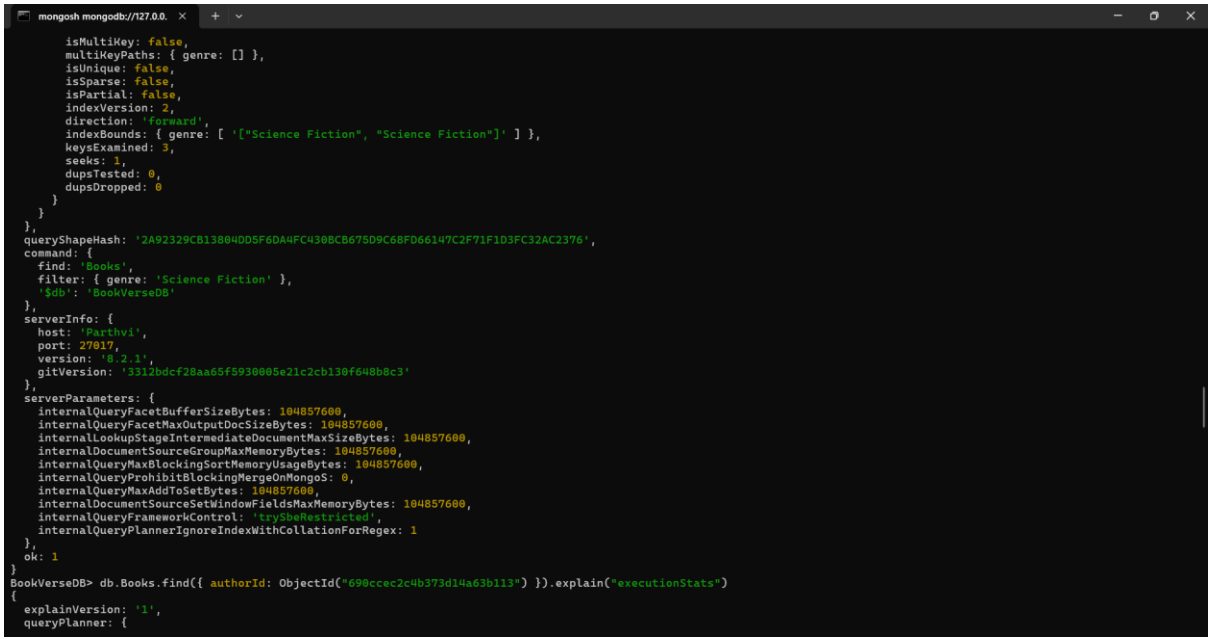
After Creating Index

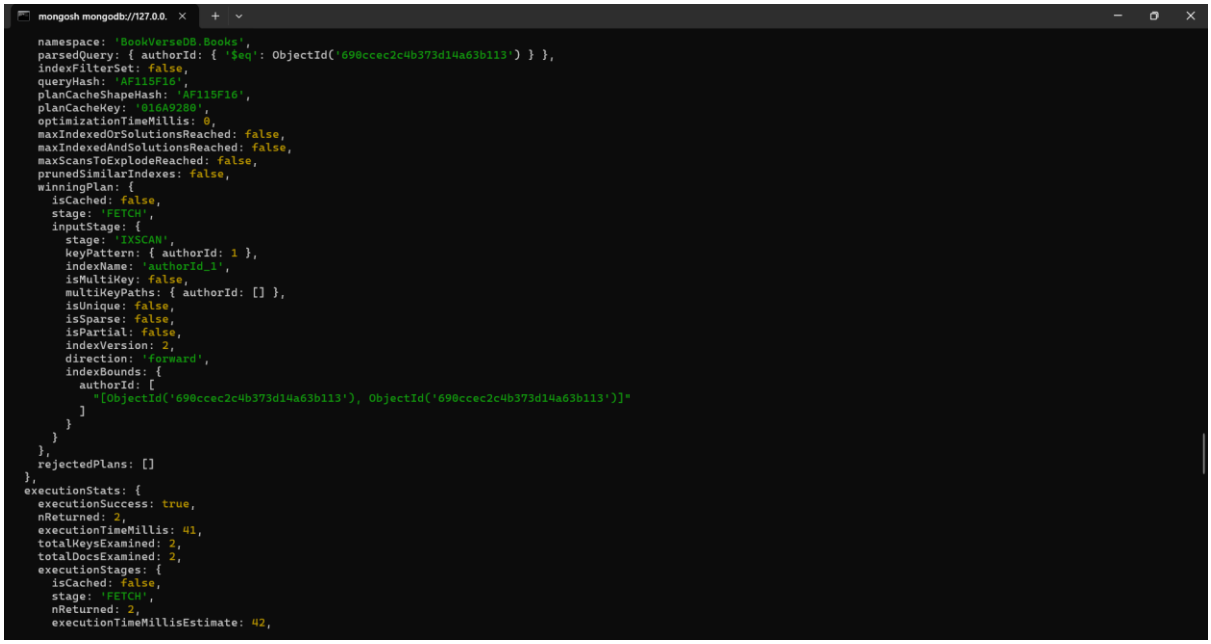


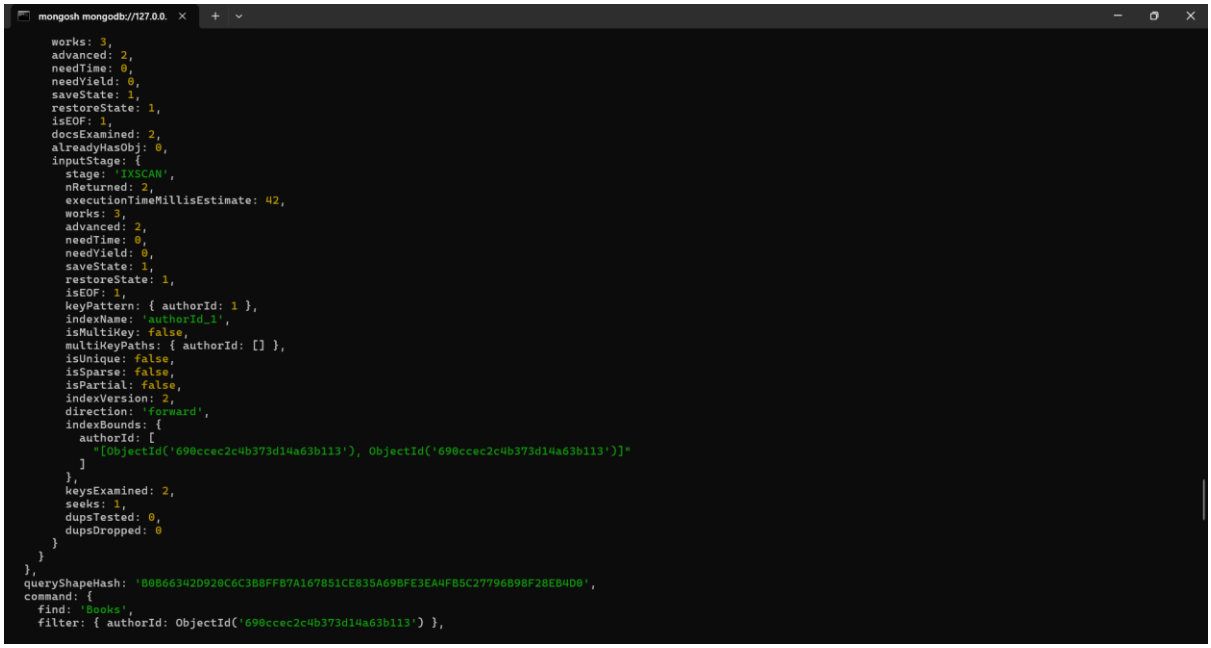
```
mongosh mongodb://127.0.0.

BookVerseDB> db.Books.createIndex({ genre: 1 })
genre_1
BookVerseDB> db.Books.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { genre: 1 }, name: 'genre_1' }
]
BookVerseDB> db.Books.createIndex({ authorId: 1 })
authorId_1
BookVerseDB> db.Books.createIndex({ "ratings.score": 1 })
ratings.score_1
BookVerseDB> db.Books.createIndex({ publicationYear: 1 })
publicationYear_1
BookVerseDB> db.Books.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { genre: 1 }, name: 'genre_1' },
  { v: 2, key: { authorId: 1 }, name: 'authorId_1' },
  { v: 2, key: { 'ratings.score': 1 }, name: 'ratings.score_1' },
  { v: 2, key: { publicationYear: 1 }, name: 'publicationYear_1' }
]
BookVerseDB> db.Books.find({ genre: "Science Fiction" }).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'BookVerseDB.Books',
    parsedQuery: { genre: { '$eq': 'Science Fiction' } },
    indexFilterSet: false,
    queryHash: 'B2E493C0',
    planCacheShapeHash: 'B2E493C0',
    planCacheKey: '30B42E12',
    optimizationTimeMillis: 41,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'FETCH',
      inputStage: {
        stage: 'IXSCAN',
        keyPattern: { genre: 1 },
        indexName: 'genre_1',
        isMultiKey: false,
        multiKeyPaths: { genre: [] },
```



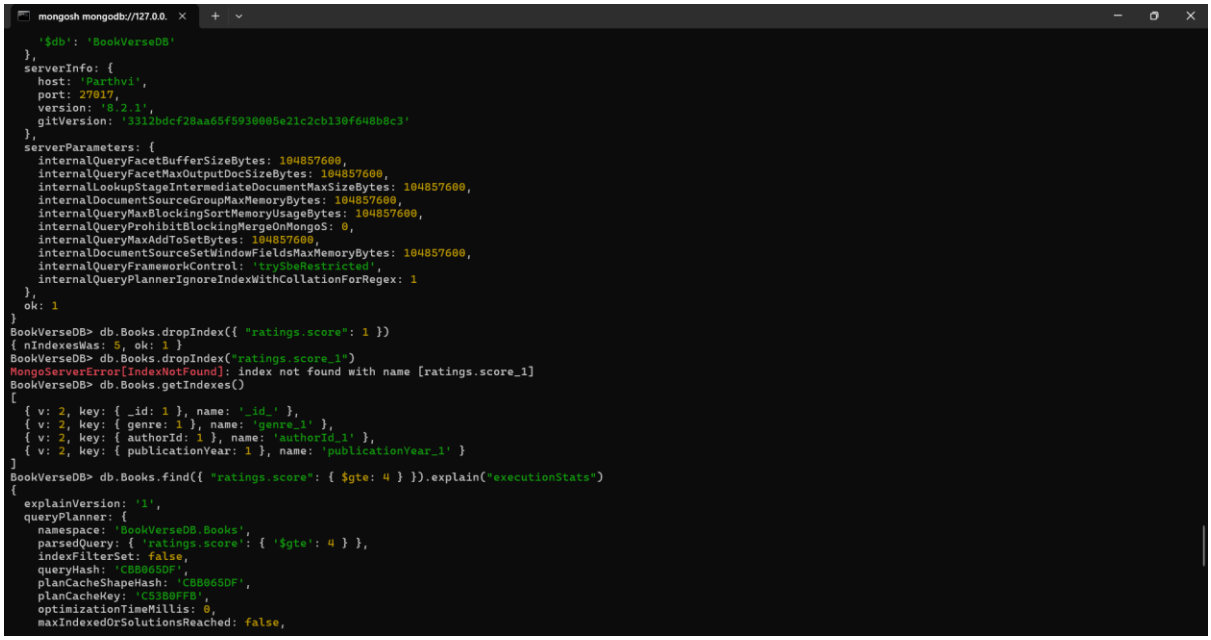
```
mongosh mongodb://127.0.0.

        multiKeyPaths: { genre: [] },
        isUnique: false,
        isSparse: false,
        isPartial: false,
        indexVersion: 2,
        direction: 'forward',
        indexBounds: { genre: [ '["Science Fiction", "Science Fiction"]' ] }
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 3,
    executionTimeMillis: 94,
    totalKeysExamined: 3,
    totalDocsExamined: 3,
    executionStages: {
      isCached: false,
      stage: 'FETCH',
      nReturned: 3,
      executionTimeMillisEstimate: 53,
      works: 4,
      advanced: 3,
      needTime: 0,
      needYield: 0,
      saveState: 1,
      restoreState: 1,
      isEOF: 1,
      docsExamined: 3,
      alreadyHasObj: 0,
      inputStage: {
        stage: 'IXSCAN',
        nReturned: 3,
        executionTimeMillisEstimate: 53,
        works: 4,
        advanced: 3,
        needTime: 0,
        needYield: 0,
        saveState: 1,
        restoreState: 1,
        isEOF: 1,
        keyPattern: { genre: 1 },
        indexName: 'genre_1',
        isMultiKey: false,
```


mongosh mongodb://127.0.0.
isMultiKey: false,
multiKeyPaths: { genre: [] },
isUnique: false,
isSparse: false,
isPartial: false,
indexVersion: 2,
direction: 'forward',
indexBounds: { genre: ['["Science Fiction", "Science Fiction"]'] },
keysExamined: 3,
seeks: 1,
dupsTested: 0,
dupsDropped: 0
queryShapeHash: '2A92329CB13804DD5F6DA4FC430BCB675D9C68FD66147C2F71F1D3FC32AC2376',
command: {
find: 'Books',
filter: { genre: 'Science Fiction' },
'$db': 'BookVerseDB'
serverInfo: {
host: 'Parthvi',
port: 27017,
version: '8.2.1',
gitVersion: '3312bdcf28aa65f5930005e21c2cb130f648b8c3'
serverParameters: {
internalQueryFacetBufferSizeBytes: 104857600,
internalQueryFacetMaxOutputDocSizeBytes: 104857600,
internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
internalDocumentSourceGroupMaxMemoryBytes: 104857600,
internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
internalQueryProhibitBlockingMergeOnMongoS: 0,
internalQueryMaxAddToSetBytes: 104857600,
internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
internalQueryFrameworkControl: 'trySbeRestricted',
internalQueryPlannerIgnoreIndexWithCollationForRegex: 1
ok: 1
BookVerseDB> db.Books.find({ authorId: ObjectId("690ccec2c4b373d14a63b113") }).explain("executionStats")
explainVersion: '1',
queryPlanner: {

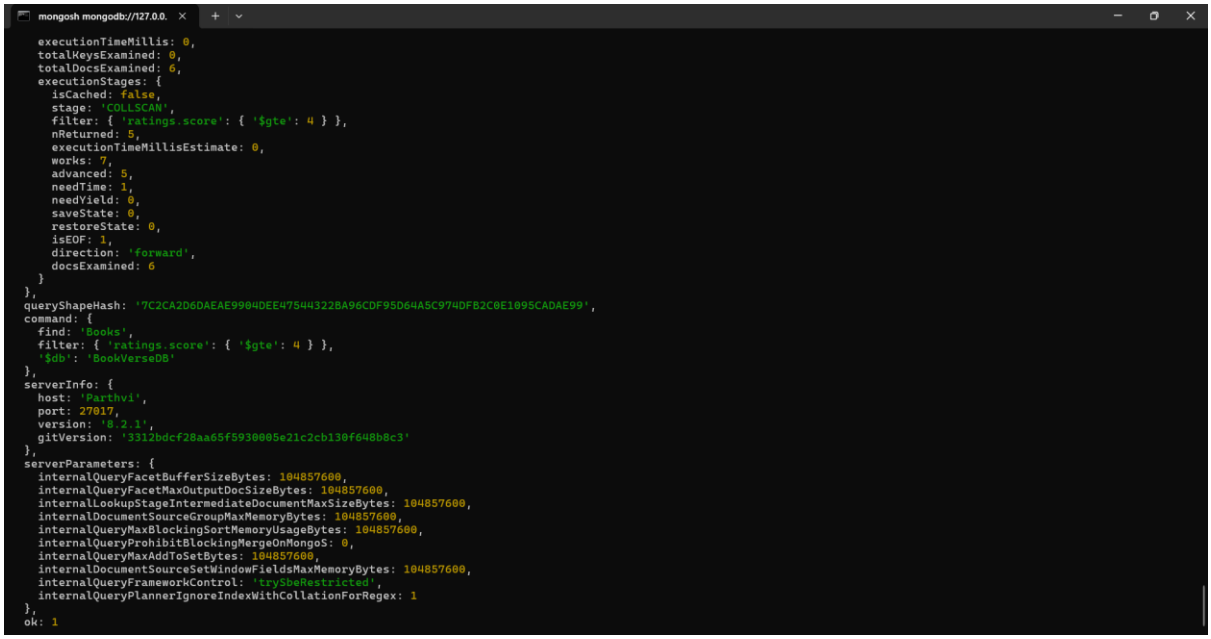

mongosh mongodb://127.0.0.
namespace: 'BookVerseDB.Books',
parsedQuery: { authorId: { '$eq': ObjectId('690ccec2c4b373d14a63b113') } },
indexFilterSet: false,
queryHash: 'AF115F16',
planCacheShapeHash: 'AF115F16',
planCacheKey: '016A9280',
optimizationTimeMillis: 0,
maxIndexedOrSolutionsReached: false,
maxIndexedAndSolutionsReached: false,
maxScansToExplodeReached: false,
prunedSimilarIndexes: false,
winningPlan: {
isCached: false,
stage: 'FETCH',
inputStage: {
stage: 'IXSCAN',
keyPattern: { authorId: 1 },
indexName: 'authorId_1',
isMultiKey: false,
multiKeyPaths: { authorId: [] },
isUnique: false,
isSparse: false,
isPartial: false,
indexVersion: 2,
direction: 'forward',
indexBounds: {
authorId: [
"[ObjectId('690ccec2c4b373d14a63b113'), ObjectId('690ccec2c4b373d14a63b113')]"
rejectedPlans: []
executionStats: {
executionSuccess: true,
nReturned: 2,
executionTimeMillis: 41,
totalKeysExamined: 2,
totalDocsExamined: 2,
executionStages: {
isCached: false,
stage: 'FETCH',
nReturned: 2,
executionTimeMillisEstimate: 42,



```
mongosh mongodb://127.0.0.
    works: 3,
    advanced: 2,
    needTime: 0,
    needYield: 0,
    saveState: 1,
    restoreState: 1,
    isEOF: 1,
    docsExamined: 2,
    alreadyHasObj: 0,
    inputStage: {
      stage: 'IXSCAN',
      nReturned: 2,
      executionTimeMillisEstimate: 42,
      works: 3,
      advanced: 2,
      needTime: 0,
      needYield: 0,
      saveState: 1,
      restoreState: 1,
      isEOF: 1,
      keyPattern: { authorId: 1 },
      indexName: 'authorId_1',
      isMultiKey: false,
      multiKeyPaths: { authorId: [] },
      isUnique: false,
      isSparse: false,
      isPartial: false,
      indexVersion: 2,
      direction: 'forward',
      indexBounds: {
        authorId: [
          "[ObjectId('690ccec2c4b373d14a63b113'), ObjectId('690ccec2c4b373d14a63b113')]"
        ]
      },
      keysExamined: 2,
      seeks: 1,
      dupsTested: 0,
      dupsDropped: 0
    }
  }
},
queryShapeHash: 'B0B66342D920C6C3B8FFB7A167851CE835A69BFE3EA4FB5C27796B98F28EB4D0',
command: {
  find: 'Books',
  filter: { authorId: ObjectId('690ccec2c4b373d14a63b113') },
```



```
mongosh mongodb://127.0.0.
  '$db': 'BookVerseDB'
},
serverInfo: {
  host: 'Parthvi',
  port: 27017,
  version: '8.2.1',
  gitVersion: '3312bdcf28aa65f5930005e21c2cb130f648b8c3'
},
serverParameters: {
  internalQueryFacetBufferSizeBytes: 104857600,
  internalQueryFacetMaxOutputDocSizeBytes: 104857600,
  internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
  internalDocumentSourceGroupMaxMemoryBytes: 104857600,
  internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
  internalQueryProhibitBlockingMergeOnMongoS: 0,
  internalQueryMaxAddToSetBytes: 104857600,
  internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
  internalQueryFrameworkControl: 'trySbeRestricted',
  internalQueryPlannerIgnoreIndexWithCollationForRegex: 1
},
ok: 1
}
BookVerseDB> db.Books.dropIndex({ "ratings.score": 1 })
{ nIndexesWas: 5, ok: 1 }
BookVerseDB> db.Books.dropIndex("ratings.score_1")
MongoServerError[IndexNotFound]: index not found with name [ratings.score_1]
BookVerseDB> db.Books.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { genre: 1 }, name: 'genre_1' },
  { v: 2, key: { authorId: 1 }, name: 'authorId_1' },
  { v: 2, key: { publicationYear: 1 }, name: 'publicationYear_1' }
]
BookVerseDB> db.Books.find({ "ratings.score": { $gte: 4 } }).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'BookVerseDB.Books',
    parsedQuery: { 'ratings.score': { '$gte': 4 } },
    indexFilterSet: false,
    queryHash: 'CBB065DF',
    planCacheShapeHash: 'CBB065DF',
    planCacheKey: 'C53B0FFB',
    optimizationTimeMillis: 0,
    maxIndexedOrSolutionsReached: false,
```



User Story 2 —

Aggregation Framework As a data analyst, I need to generate reports about books and their ratings using MongoDB's aggregation pipeline.

Tasks:

1. Calculate the average rating per book using $unwind, $group, and $avg.

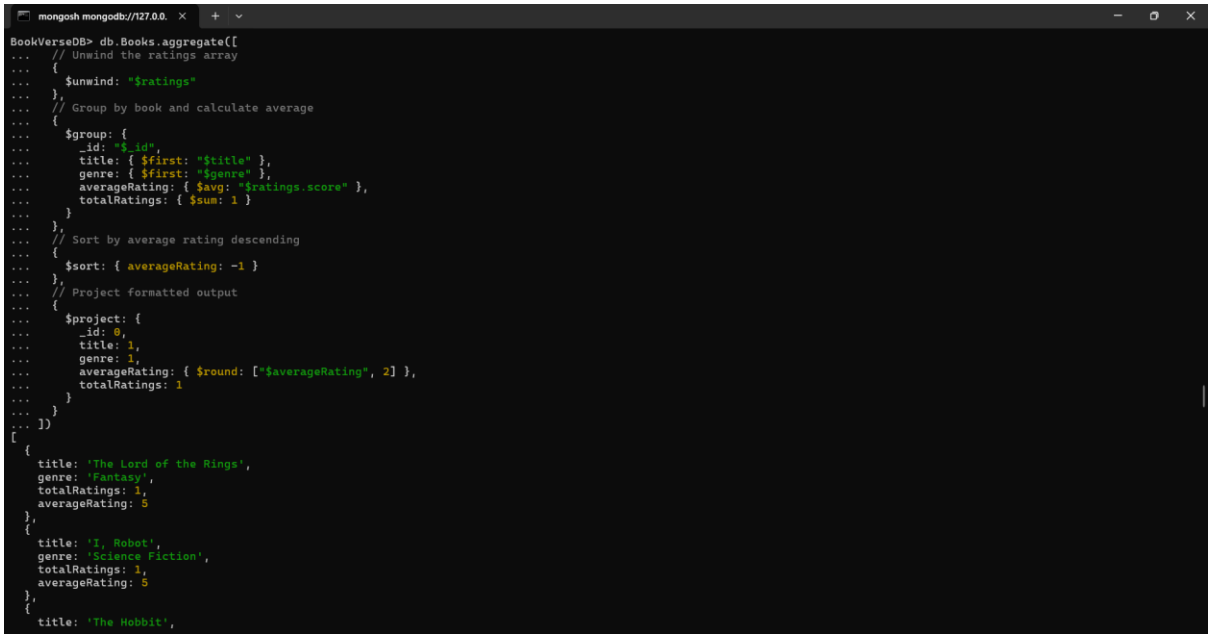
2. Retrieve the top 3 highest-rated books.

3. Count the number of books published per genre.

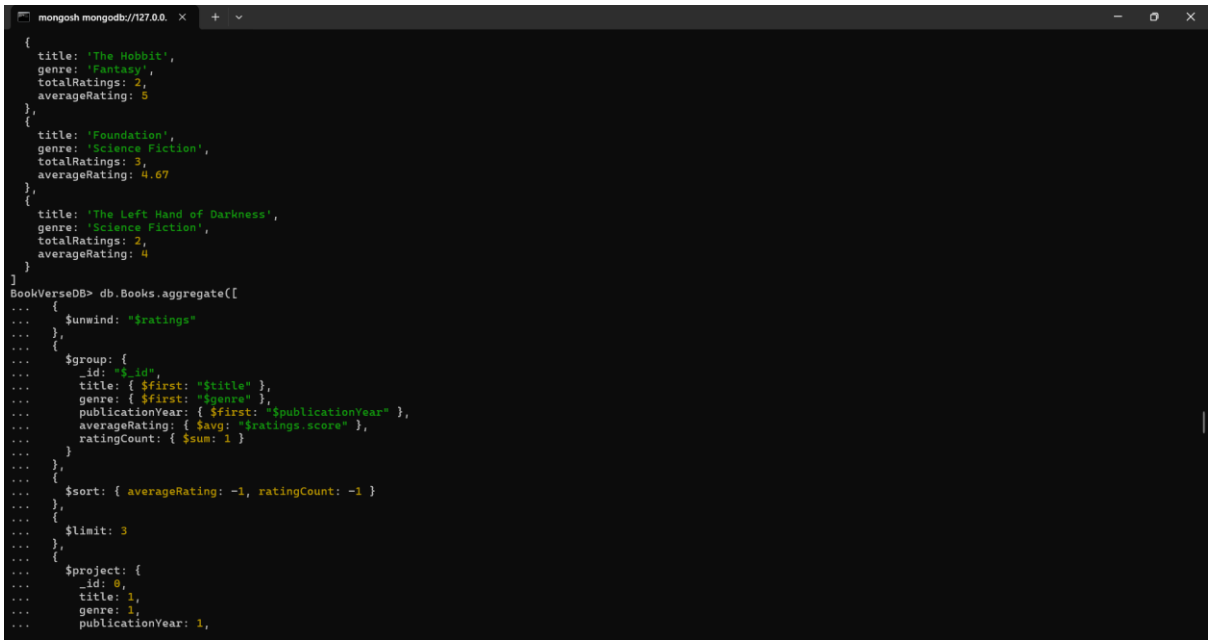
4. Find authors who have more than 2 books published.

5. Display the total reward points (sum of all ratings) received by each author.

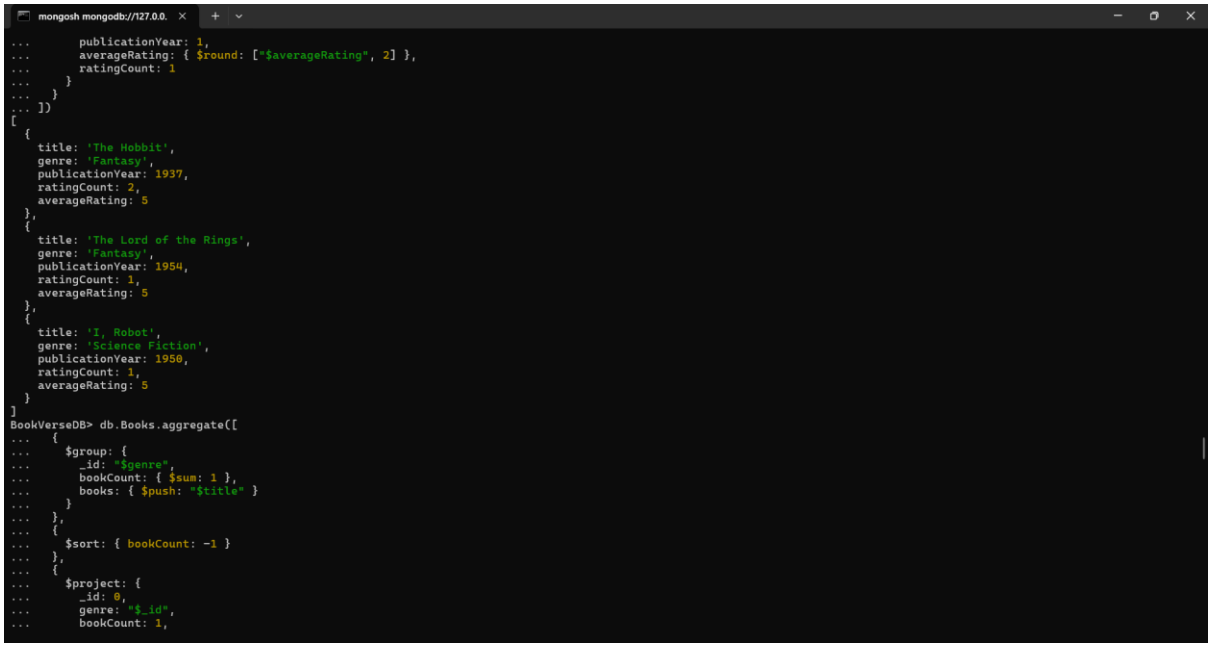
Concepts Covered: Aggregation stages, grouping, projection, sorting



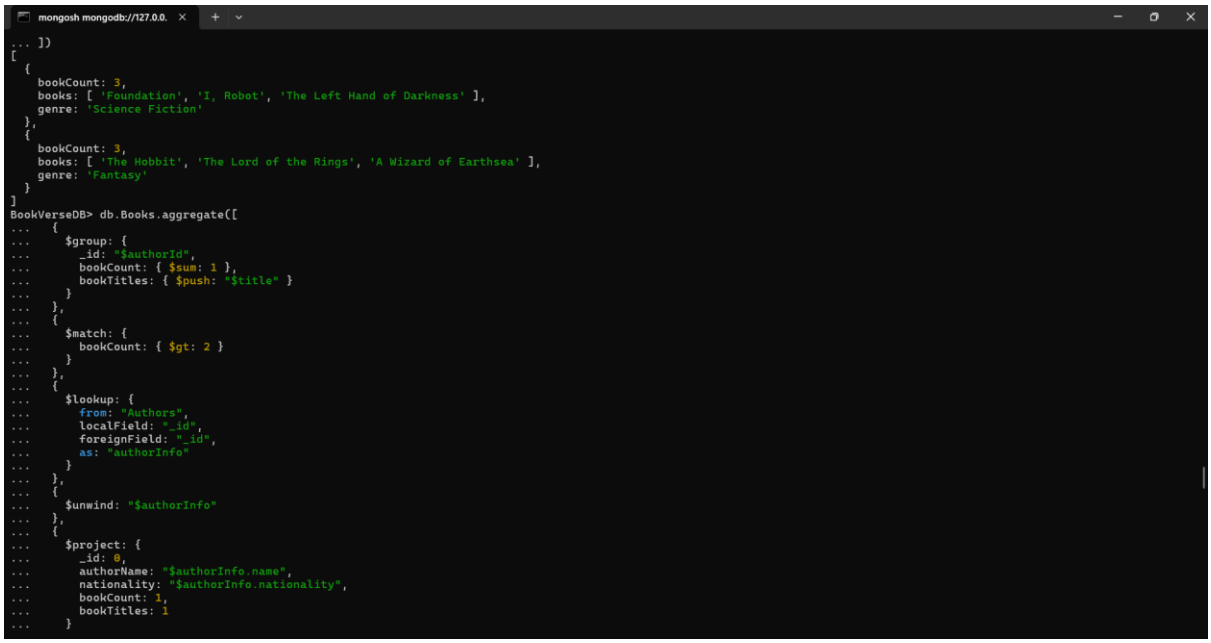
```
mongosh mongodb://127.0.0.

BookVerseDB> db.Books.aggregate([
...   // Unwind the ratings array
...   {
...     $unwind: "$ratings"
...   },
...   // Group by book and calculate average
...   {
...     $group: {
...       _id: "$_id",
...       title: { $first: "$title" },
...       genre: { $first: "$genre" },
...       averageRating: { $avg: "$ratings.score" },
...       totalRatings: { $sum: 1 }
...     }
...   },
...   // Sort by average rating descending
...   {
...     $sort: { averageRating: -1 }
...   },
...   // Project formatted output
...   {
...     $project: {
...       _id: 0,
...       title: 1,
...       genre: 1,
...       averageRating: { $round: ["$averageRating", 2] },
...       totalRatings: 1
...     }
...   }
... ])
[
  {
    title: 'The Lord of the Rings',
    genre: 'Fantasy',
    totalRatings: 1,
    averageRating: 5
  },
  {
    title: 'I, Robot',
    genre: 'Science Fiction',
    totalRatings: 1,
    averageRating: 5
  },
  {
    title: 'The Hobbit',
```



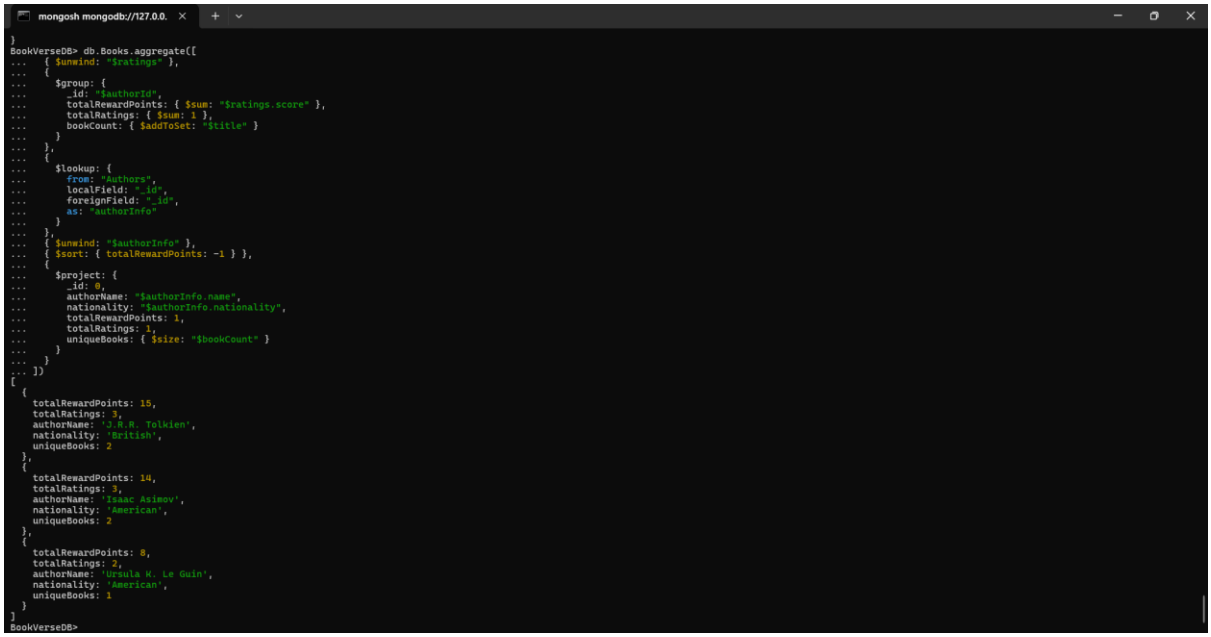
```
mongosh mongodb://127.0.0.

  {
    title: 'The Hobbit',
    genre: 'Fantasy',
    totalRatings: 2,
    averageRating: 5
  },
  {
    title: 'Foundation',
    genre: 'Science Fiction',
    totalRatings: 3,
    averageRating: 4.67
  },
  {
    title: 'The Left Hand of Darkness',
    genre: 'Science Fiction',
    totalRatings: 2,
    averageRating: 4
  }
]
BookVerseDB> db.Books.aggregate([
...   {
...     $unwind: "$ratings"
...   },
...   {
...     $group: {
...       _id: "$_id",
...       title: { $first: "$title" },
...       genre: { $first: "$genre" },
...       publicationYear: { $first: "$publicationYear" },
...       averageRating: { $avg: "$ratings.score" },
...       ratingCount: { $sum: 1 }
...     }
...   },
...   {
...     $sort: { averageRating: -1, ratingCount: -1 }
...   },
...   {
...     $limit: 3
...   },
...   {
...     $project: {
...       _id: 0,
...       title: 1,
...       genre: 1,
...       publicationYear: 1,
```



```
mongosh mongodb://127.0.0.
...       publicationYear: 1,
...       averageRating: { $round: ["$averageRating", 2] },
...       ratingCount: 1
...     }
...   }
... ])
[
  {
    title: 'The Hobbit',
    genre: 'Fantasy',
    publicationYear: 1937,
    ratingCount: 2,
    averageRating: 5
  },
  {
    title: 'The Lord of the Rings',
    genre: 'Fantasy',
    publicationYear: 1954,
    ratingCount: 1,
    averageRating: 5
  },
  {
    title: 'I, Robot',
    genre: 'Science Fiction',
    publicationYear: 1950,
    ratingCount: 1,
    averageRating: 5
  }
]
BookVerseDB> db.Books.aggregate([
...   {
...     $group: {
...       _id: "$genre",
...       bookCount: { $sum: 1 },
...       books: { $push: "$title" }
...     }
...   },
...   {
...     $sort: { bookCount: -1 }
...   },
...   {
...     $project: {
...       _id: 0,
...       genre: "$_id",
...       bookCount: 1,
```



```
mongosh mongodb://127.0.0.
... ])
[
  {
    bookCount: 3,
    books: [ 'Foundation', 'I, Robot', 'The Left Hand of Darkness' ],
    genre: 'Science Fiction'
  },
  {
    bookCount: 3,
    books: [ 'The Hobbit', 'The Lord of the Rings', 'A Wizard of Earthsea' ],
    genre: 'Fantasy'
  }
]
BookVerseDB> db.Books.aggregate([
...   {
...     $group: {
...       _id: "$authorId",
...       bookCount: { $sum: 1 },
...       bookTitles: { $push: "$title" }
...     }
...   },
...   {
...     $match: {
...       bookCount: { $gt: 2 }
...     }
...   },
...   {
...     $lookup: {
...       from: "Authors",
...       localField: "_id",
...       foreignField: "_id",
...       as: "authorInfo"
...     }
...   },
...   {
...     $unwind: "$authorInfo"
...   },
...   {
...     $project: {
...       _id: 0,
...       authorName: "$authorInfo.name",
...       nationality: "$authorInfo.nationality",
...       bookCount: 1,
...       bookTitles: 1
...     }
```



User Story 3 —

MongoDB Atlas Connection As a backend engineer, I need to deploy and connect the database to MongoDB Atlas for cloud management.

Tasks:

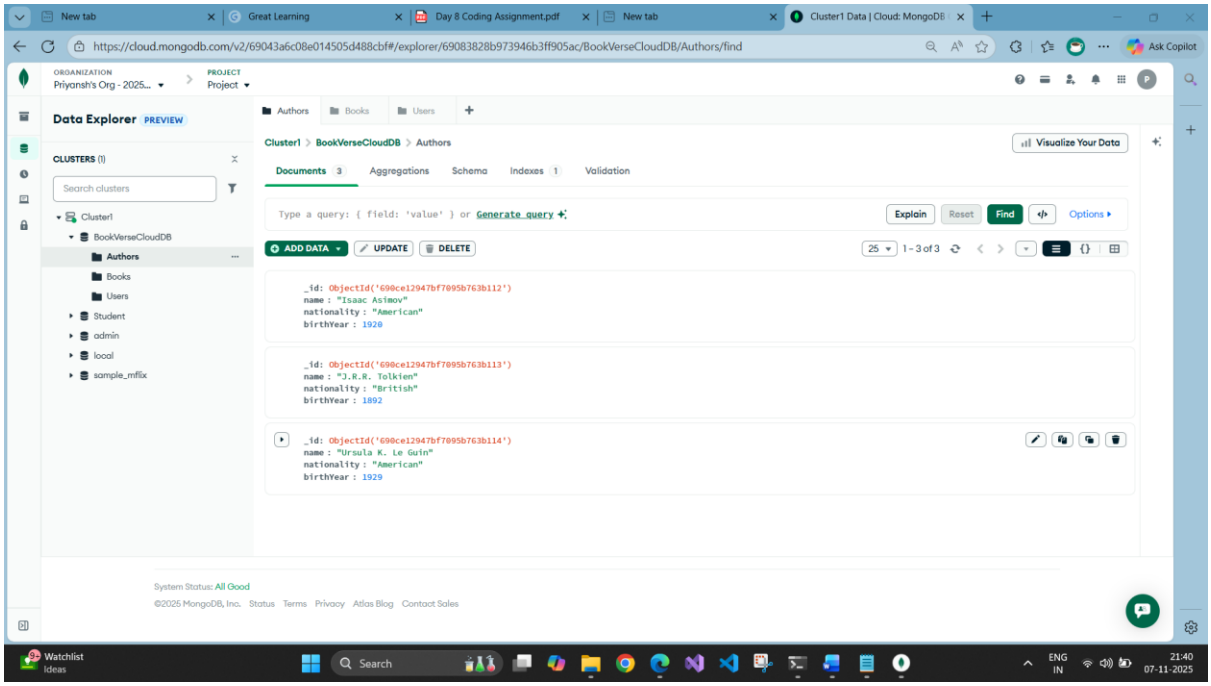
1. Create a free cluster in MongoDB Atlas.

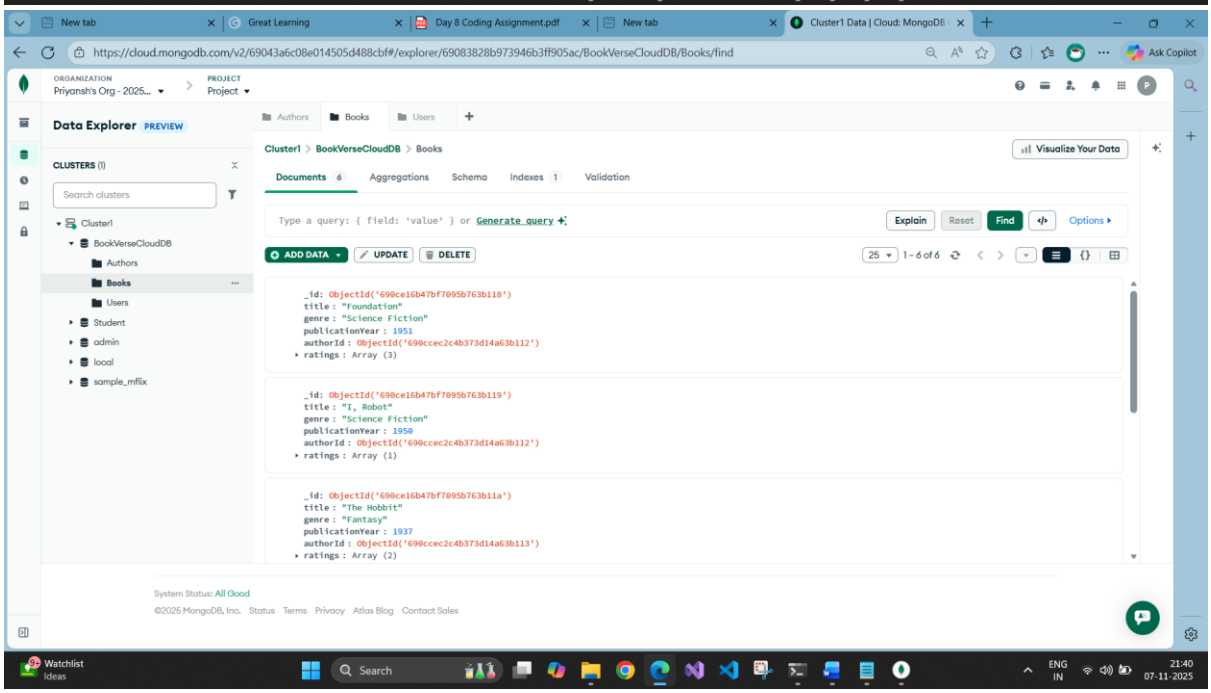
2. Create a database named BookVerseCloudDB.

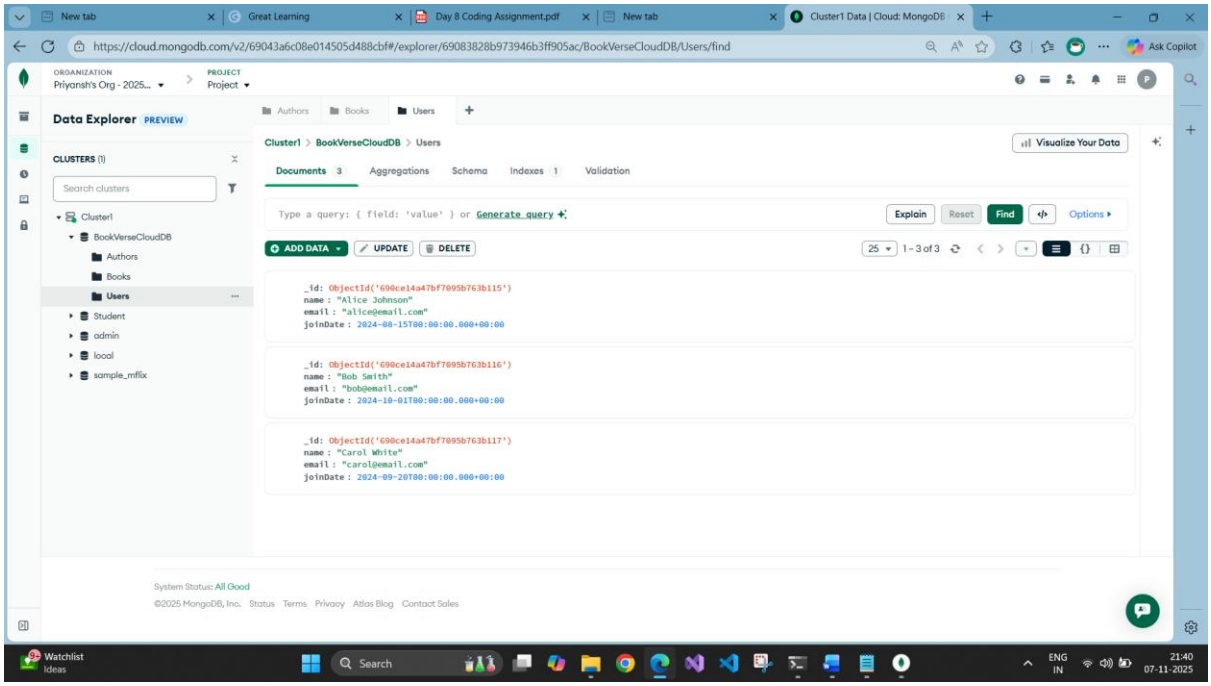
3. Import your local collections (Authors, Books, Users) into Atlas.

4. Connect to your cluster using the connection string in a Node.js script or MongoDB Compass.

Concepts Covered: Cloud database setup, connection configuration, data import


New tab
Great Learning
Day 8 Coding Assignment.pdf
New tab
Cluster1 Data | Cloud: MongoDB
https://cloud.mongodb.com/v2/69043a6c08e014505d488cbf#/explorer/69083828b973946b3ff905ac/BookVerseCloudDB/Authors/find
Ask Copilot
ORGANIZATION
Priyansh's Org - 2025...
PROJECT
Project
Data Explorer PREVIEW
CLUSTERS (1)
Search clusters
Cluster1
BookVerseCloudDB
Authors
Books
Users
Student
admin
local
sample_mflix
Authors
Books
Users
Cluster1 > BookVerseCloudDB > Authors
Visualize Your Data
Documents 3
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
UPDATE
DELETE
25
1 - 3 of 3
_id: ObjectId('690ce12947bf7095b763b112')
name : "Isaac Asimov"
nationality : "American"
birthYear : 1920
_id: ObjectId('690ce12947bf7095b763b113')
name : "J.R.R. Tolkien"
nationality : "British"
birthYear : 1892
_id: ObjectId('690ce12947bf7095b763b114')
name : "Ursula K. Le Guin"
nationality : "American"
birthYear : 1929
System Status: All Good
©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales
Watchlist
Ideas
Search
ENG
IN
21:40
07-11-2025


New tab
Great Learning
Day 8 Coding Assignment.pdf
New tab
Cluster1 Data | Cloud: MongoDB
https://cloud.mongodb.com/v2/69043a6c08e014505d488cbf#/explorer/69083828b973946b3ff905ac/BookVerseCloudDB/Books/find
Ask Copilot
ORGANIZATION
Priyansh's Org - 2025...
PROJECT
Project
Data Explorer PREVIEW
CLUSTERS (1)
Search clusters
Cluster1
BookVerseCloudDB
Authors
Books
Users
Student
admin
local
sample_mflix
Authors
Books
Users
Cluster1 > BookVerseCloudDB > Books
Visualize Your Data
Documents 6
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
UPDATE
DELETE
25
1 - 6 of 6
_id: ObjectId('690ce16b47bf7095b763b118')
title : "Foundation"
genre : "Science Fiction"
publicationYear : 1951
authorId : ObjectId('690ccec2c4b373d14a63b112')
ratings : Array (3)
_id: ObjectId('690ce16b47bf7095b763b119')
title : "I, Robot"
genre : "Science Fiction"
publicationYear : 1950
authorId : ObjectId('690ccec2c4b373d14a63b112')
ratings : Array (1)
_id: ObjectId('690ce16b47bf7095b763b11a')
title : "The Hobbit"
genre : "Fantasy"
publicationYear : 1937
authorId : ObjectId('690ccec2c4b373d14a63b113')
ratings : Array (2)
System Status: All Good
©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales
Watchlist
Ideas
Search
ENG
IN
21:40
07-11-2025


New tab
Great Learning
Day 8 Coding Assignment.pdf
New tab
Cluster1 Data | Cloud: MongoDB
https://cloud.mongodb.com/v2/69043a6c08e014505d488cbf#/explorer/69083828b973946b3ff905ac/BookVerseCloudDB/Users/find
Ask Copilot
ORGANIZATION
Priyansh's Org - 2025...
PROJECT
Project
Data Explorer PREVIEW
CLUSTERS (1)
Search clusters
Cluster1
BookVerseCloudDB
Authors
Books
Users
Student
admin
local
sample_mflix
Authors
Books
Users
Cluster1 > BookVerseCloudDB > Users
Visualize Your Data
Documents 3
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
UPDATE
DELETE
25
1 - 3 of 3
_id: ObjectId('690ce14a47bf7095b763b115')
name : "Alice Johnson"
email : "alice@email.com"
joinDate : 2024-08-15T00:00:00.000+00:00
_id: ObjectId('690ce14a47bf7095b763b116')
name : "Bob Smith"
email : "bob@email.com"
joinDate : 2024-10-01T00:00:00.000+00:00
_id: ObjectId('690ce14a47bf7095b763b117')
name : "Carol White"
email : "carol@email.com"
joinDate : 2024-09-20T00:00:00.000+00:00
System Status: All Good
©2025 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales
Watchlist
Ideas
Search
ENG
IN
21:40
07-11-2025