Task 6: Advanced Indexing in MongoDB

Objective:

Understand and practice advanced indexing techniques in MongoDB, including creating and using compound indexes, understanding index properties (unique, sparse), and creating text indexes.

Prerequisites:

- Basic understanding of JavaScript and MongoDB.
- Node.js installation.
- MongoDB installed and running.
- A MongoDB collection with sample movie data.

Concepts:

1. **Advanced Indexing:**

   Compound Indexes:

   - Compound indexes are indexes on multiple fields.
   - They support queries that match on multiple fields and can be used to sort results.

     Example:

     JavaScript:

     ```
     const compoundIndex = await collection.createIndex({ title: 1, release_date: -1 });
     console.log('Compound Index:', compoundIndex);
     ```

     MongoDB Compass:

     - Open MongoDB Compass.
     - Connect to your MongoDB instance.
     - Select your database and collection.
     - Click on the `Indexes` tab.
     - Click on `Create Index`.
     - Add the fields for the index:

       ```
       {
         "title": 1,
         "release_date": -1
       }
       ```

     - Click on the `Create Index` button.

   Index Properties (Unique, Sparse):

   Unique Index:

- Ensures that the indexed field does not store duplicate values.

  Example:

  JavaScript:

  ```
  const uniqueIndex = await collection.createIndex({ title: 1 },
  { unique: true });
  console.log('Unique Index:', uniqueIndex);
  ```

  MongoDB Compass:

  - Follow the same steps to navigate to your collection in MongoDB Compass.
  - Click on the `Indexes` tab.
  - Click on `Create Index`.
  - Add the field and select `Unique` for the unique index:

    ```
    {
      "title": 1
    }
    ```

  - Check the `Unique` option.
  - Click on the `Create Index` button.

Sparse Index:

- Only indexes the documents that contain the indexed field.

  Example:

  JavaScript:

  ```
  const sparseIndex = await collection.createIndex({ video: 1 },
  { sparse: true });
  console.log('Sparse Index:', sparseIndex);
  ```

  MongoDB Compass:

  - For the sparse index, add the field and select `Sparse`:

    ```
    {
      "video": 1
    }
    ```

  - Check the `Sparse` option.
  - Click on the `Create Index` button.

Text Indexes:

- Text indexes support text search queries on string content.
- They can index multiple fields and support language-specific features.

  Example:

JavaScript:

```
const textIndex = await collection.createIndex({ title: 'text', overview:
'text' });
console.log('Text Index:', textIndex);
```

MongoDB Compass:

- Follow the same steps to navigate to your collection in MongoDB Compass.
- Click on the `Indexes` tab.
- Click on `Create Index` .
- Add the fields and select `text` for the text index:

  ```
  {
    "title": "text",
    "overview": "text"
  }
  ```

- Click on the `Create Index` button.

Instructions:

Perform the following queries:

1. Create a compound index on the genre and release_date fields.
2. Create a unique index on the id field.
3. Create a sparse index on the backdrop_path field.
4. Create a text index on the original_title and overview fields.
5. Use the compound index to find movies in a specific genre released after a specific date.
6. Use the unique index to attempt inserting a document with a duplicate id to see the uniqueness constraint.
7. Use the sparse index to query movies that have the backdrop_path field.
8. Use the text index to perform a text search for movies with a specific keyword in their title or overview.

Resources:

- **Documentation:**
  - Indexes
    Indexes support efficient execution of queries in MongoDB. Without — indexes, Mon...
    https://docs.mongodb.com/manual/indexes/
  - Compound Indexes
    Compound indexes collect and sort data from two or more fields in each — docume...
    https://docs.mongodb.com/manual/core/index-compound/

  - Unique Indexes
    A unique index ensures that the indexed fields do not store duplicate — values; i.e. e...
    https://docs.mongodb.com/manual/core/index-unique/
  - Sparse Indexes
    Sparse indexes only contain entries for documents that have the indexed — field, ev...
    https://docs.mongodb.com/manual/core/index-sparse/
  - Text Indexes
    This page describes text query capabilities for self-managed — (non-Atlas) deploym...
    https://docs.mongodb.com/manual/core/index-text/
- **Videos:**
  - MongoDB Courses and Trainings | MongoDB University
    Discover our MongoDB Database Management courses and begin improving your CV with Mong...
    https://university.mongodb.com/courses/M201/about
  -

    Net Ninja 6:44min 77,776 Views 1,470 Likes

created with craft

-

  Cognitive Programmer 9:18 min 17,818 Views 329 Likes

GitHub Instructions

Open in Visual Studio Code:

After clicking on the "Open in Visual Studio Code" button from the GitHub Classroom confirmation page, Visual Studio Code (VSCode) will open the repository directly.

If prompted, select "Open" or "Allow" to open the repository in VSCode.

Complete the Task:

In VSCode, open the `index.js` file in the root directory of your repository and write your solution.

Ensure the `package.json` file is present and contains all necessary dependencies. If you need to install additional packages, use:

```
npm i
```

Run and Test Your Code:

Run your code to ensure it works correctly. Use the following command:

```
node index.js
```

Commit Your Changes:

Commit your changes with a meaningful message:

```
git commit -m "Completed task 6"
```

Push Your Changes to Your Forked Repository:

Push your changes to your forked repository:

```
git push origin main
```

Create a Pull Request:

Go to your forked repository on GitHub.

Click on the "Pull Requests" tab.

Click the "New Pull Request" button.

Ensure the base repository is the original template repository and the base branch is `main` .

Ensure the head repository is your forked repository and the compare branch is `main` .

Click "Create Pull Request".

Add a title and description for your pull request and submit it.

Summary of Commands

```
# Fork the repository on GitHub

# Clone the forked repository
git clone https://github.com/your-github-username/repository-name.git
cd repository-name

# Complete the task by editing index.js

# Run your code
node index.js

# Add, commit, and push your changes
git commit -m "Completed task 6"
git push origin main

# Create a pull request on GitHub
```