Building Web-based Applications with MERNStack

Milestone: Implementation in NoSQL

Group 13

Harini Prasad Vasisht +1(984)374-4836

Sushmitha Sudharsan +1(857)565-8800

vasisht.h@northeastern.edu

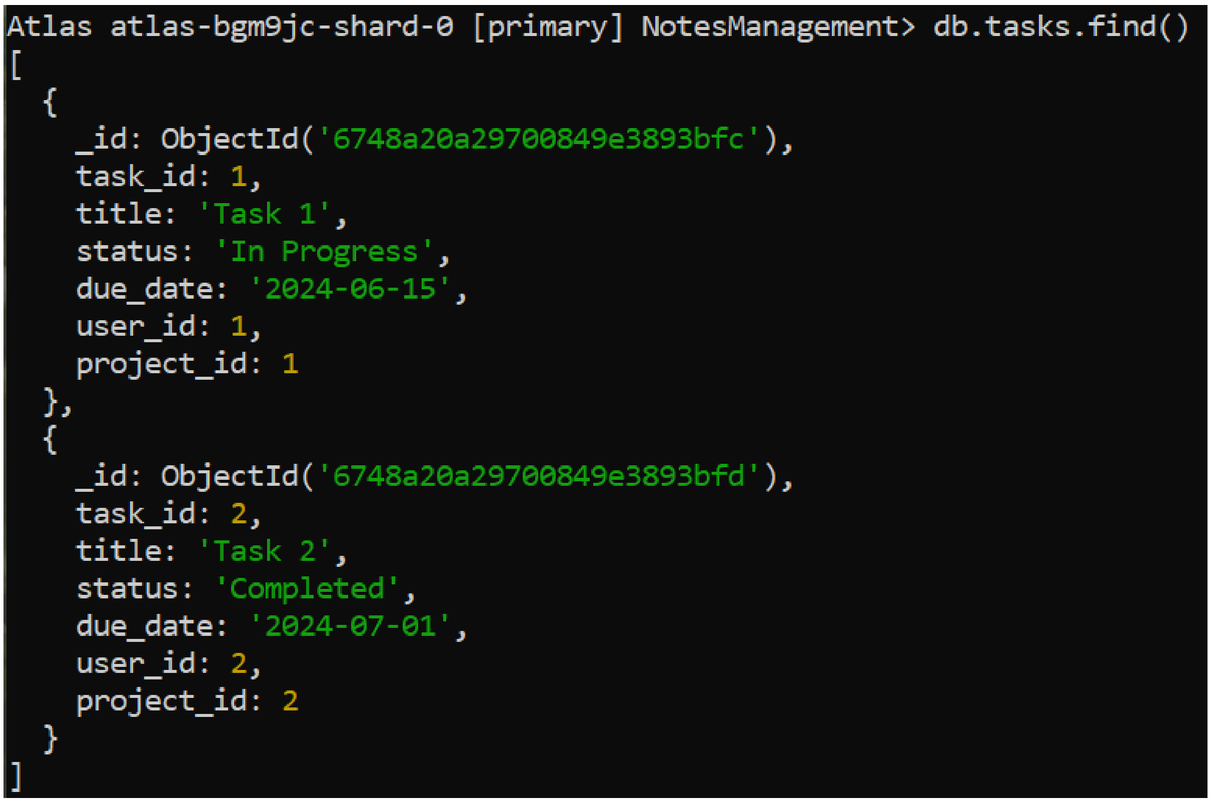
sudharsan.s@northeastern.edu

Percentage of Effort Contributed by Student1: 50%

Percentage of Effort Contributed by Student2: 50%

Signature of Student 1: harini

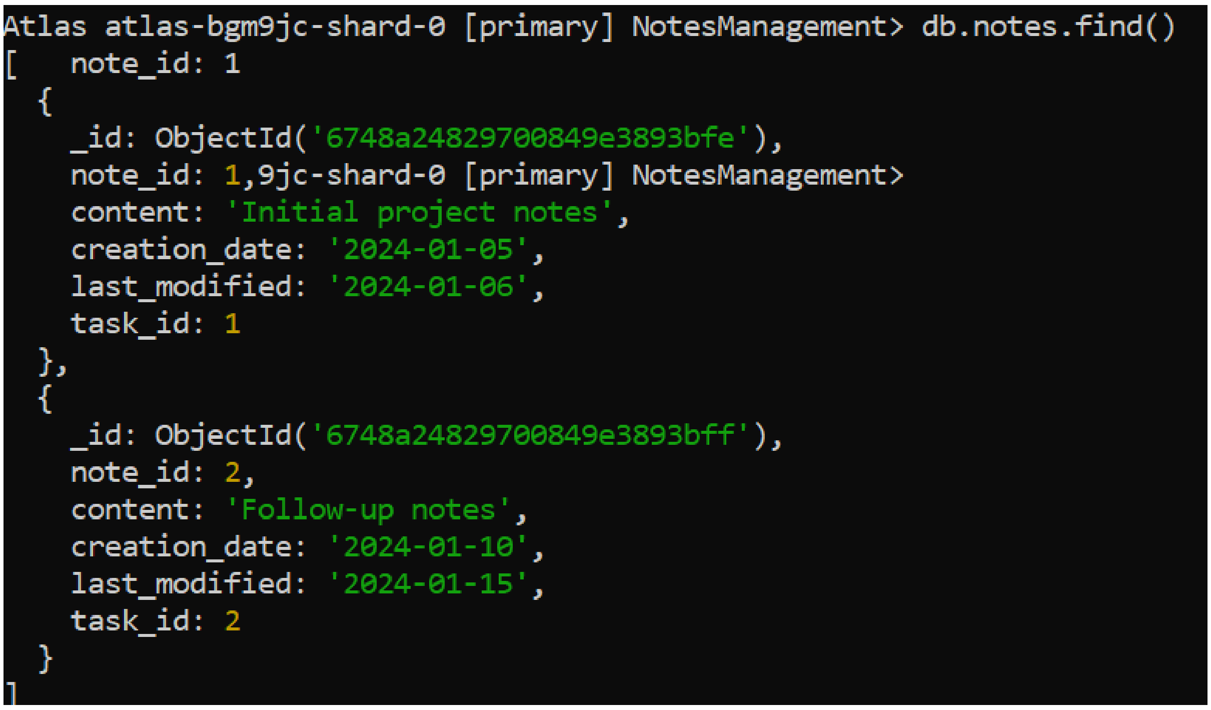
Signature of Student 2: Sushmitha



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.tasks.find()
[
  {
    _id: ObjectId('6748a20a29700849e3893bfc'),
    task_id: 1,
    title: 'Task 1',
    status: 'In Progress',
    due_date: '2024-06-15',
    user_id: 1,
    project_id: 1
  },
  {
    _id: ObjectId('6748a20a29700849e3893bfd'),
    task_id: 2,
    title: 'Task 2',
    status: 'Completed',
    due_date: '2024-07-01',
    user_id: 2,
    project_id: 2
  }
]
```

The tasks collection in the NotesManagement database contains details about various tasks, including their statuses, due dates, and associations with users and projects. Below are the key attributes and their descriptions as observed from the query output (db.tasks.find()):

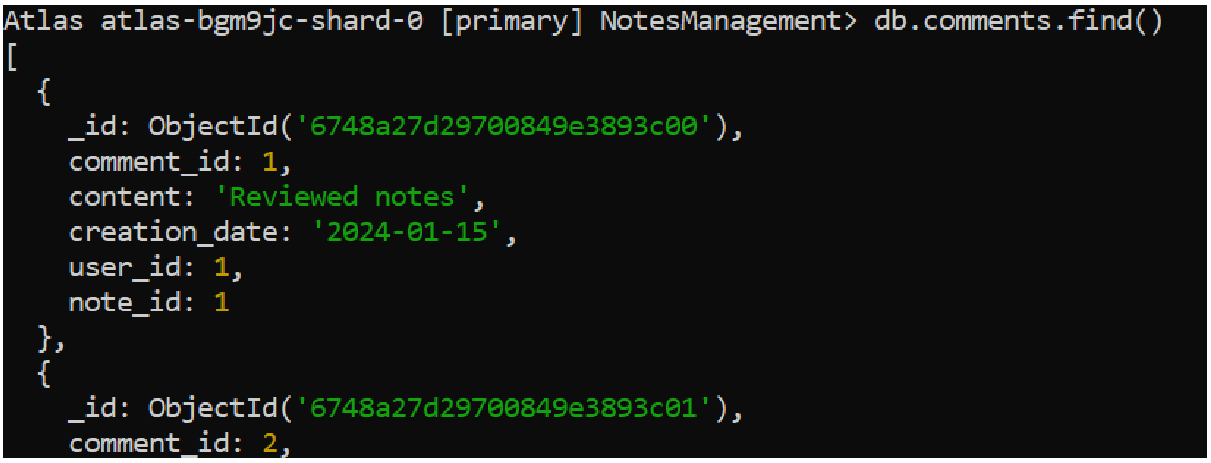
- **Attributes**:
  - _id: A unique identifier (ObjectId) automatically generated by MongoDB for each document.
  - task_id: A unique identifier for each task within the system.
  - title: The title or name of the task.
  - status: The current status of the task, such as "In Progress" or "Completed."
  - due_date: The date by which the task is expected to be completed.
  - user_id: A reference to the user assigned to the task.
  - project_id: A reference to the project to which the task belongs.



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.notes.find()
[   note_id: 1
  {
    _id: ObjectId('6748a24829700849e3893bfe'),
    note_id: 1,9jc-shard-0 [primary] NotesManagement>
    content: 'Initial project notes',
    creation_date: '2024-01-05',
    last_modified: '2024-01-06',
    task_id: 1
  },
  {
    _id: ObjectId('6748a24829700849e3893bff'),
    note_id: 2,
    content: 'Follow-up notes',
    creation_date: '2024-01-10',
    last_modified: '2024-01-15',
    task_id: 2
  }
]
```

The notes collection in the NotesManagement database contains details about various notes associated with tasks. Below are the key attributes and their descriptions as observed from the query output (db.notes.find()):

- **Attributes**:
  - _id: A unique identifier (ObjectId) automatically generated by MongoDB for each document.
  - note_id: A unique identifier for each note within the system.
  - content: Text content of the note, which describes or documents information related to the task.
  - creation_date: The date on which the note was created.
  - last_modified: The date on which the note was last updated.
  - task_id: A reference to the task that this note is associated with.



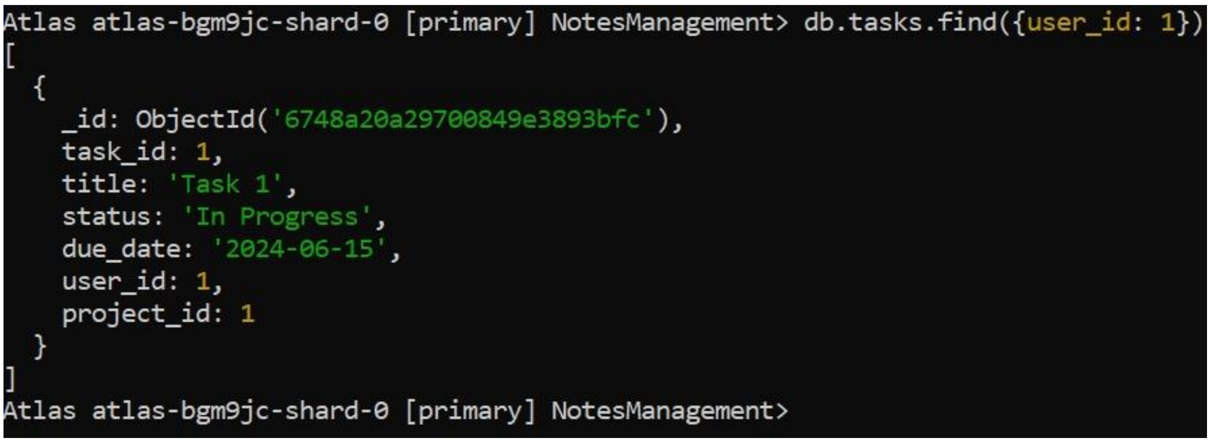
```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.comments.find()
[
  {
    _id: ObjectId('6748a27d29700849e3893c00'),
    comment_id: 1,
    content: 'Reviewed notes',
    creation_date: '2024-01-15',
    user_id: 1,
    note_id: 1
  },
  {
    _id: ObjectId('6748a27d29700849e3893c01'),
    comment_id: 2,
```

Comments Collection Query
The query db.comments.find() is used to retrieve all documents from the comments collection in the NotesManagement database. The purpose of this query is to fetch and display information about the comments associated with notes in the system.

Key Attributes:

- _id: A unique identifier (ObjectId) generated by MongoDB for each comment document.
- comment_id: A unique numeric identifier for each comment within the system.
- content: The text content of the comment, detailing the input or review provided.
- creation_date: The date on which the comment was created.
- user_id: A reference to the user who created the comment.
- note_id: A reference to the note that the comment is associated with.

Summary:
This query retrieves and displays all the comments stored in the comments collection. Each document contains detailed information about individual comments, including their association with specific users and notes, facilitating efficient tracking and organization of feedback or reviews related to the note-taking system.



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.tasks.find({user_id: 1})
[
  {
    _id: ObjectId('6748a20a29700849e3893bfc'),
    task_id: 1,
    title: 'Task 1',
    status: 'In Progress',
    due_date: '2024-06-15',
    user_id: 1,
    project_id: 1
  }
]
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement>
```

Explanation of the Query

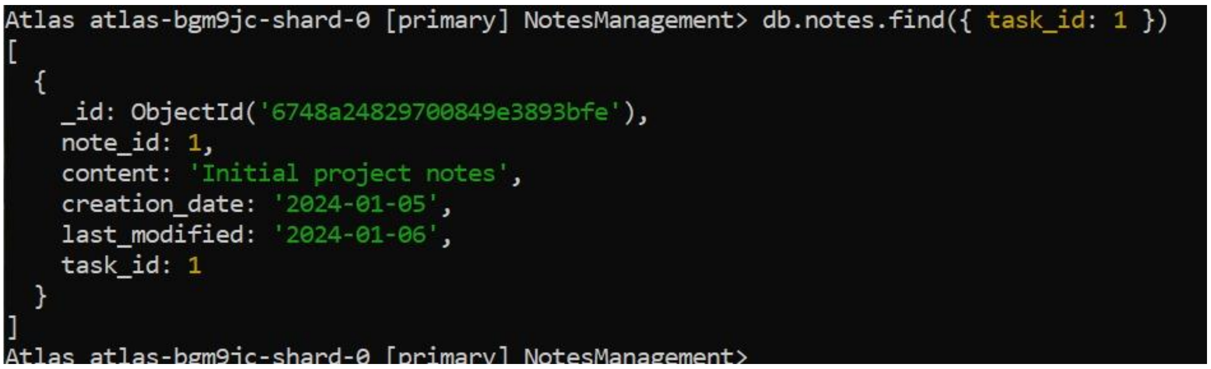
The query db.tasks.find({user_id: 1}) retrieves all documents from the tasks collection in the NotesManagement database where the user_id field equals 1. This is a filtered query, meaning it only fetches tasks assigned to the user with user_id 1.

Query Purpose:

The purpose of this query is to identify and display tasks that are specifically associated with a particular user (in this case, the user with ID 1). It enables targeted data retrieval for personalized task management or user-specific analytics.

Summary:

This query efficiently narrows down tasks in the collection to those assigned to a specific user. It is useful for scenarios where task data needs to be filtered by the responsible user, ensuring personalized task tracking and management.



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.notes.find({ task_id: 1 })
[
  {
    _id: ObjectId('6748a24829700849e3893bfe'),
    note_id: 1,
    content: 'Initial project notes',
    creation_date: '2024-01-05',
    last_modified: '2024-01-06',
    task_id: 1
  }
]
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement>
```

Explanation of the Query

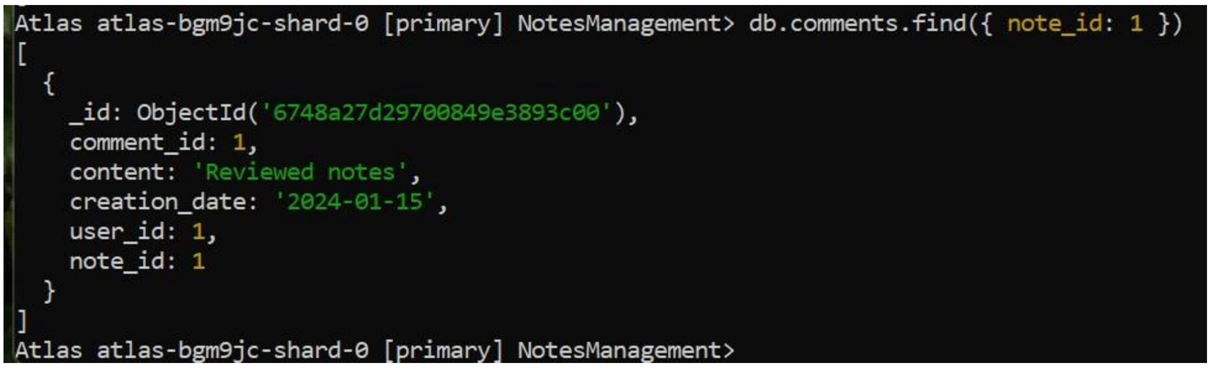
The query db.notes.find({task_id: 1}) retrieves all documents from the notes collection in the NotesManagement database where the task_id field equals 1. This is a filtered query, meaning it only fetches notes associated with the task that has task_id 1.

Query Purpose:

The purpose of this query is to identify and display notes that are specifically linked to a particular task (in this case, the task with ID 1). It enables targeted data retrieval for task-specific note tracking or task-related documentation.

Summary:

This query efficiently narrows down notes in the collection to those associated with a specific task. It is useful for scenarios where note data needs to be filtered by the task it pertains to, ensuring detailed tracking and management of task-related information.



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.comments.find({ note_id: 1 })
[
  {
    _id: ObjectId('6748a27d29700849e3893c00'),
    comment_id: 1,
    content: 'Reviewed notes',
    creation_date: '2024-01-15',
    user_id: 1,
    note_id: 1
  }
]
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement>
```

Explanation of the Query

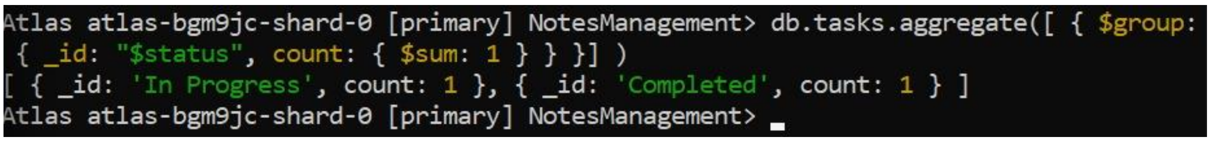
The query db.comments.find({note_id: 1}) retrieves all documents from the comments collection in the NotesManagement database where the note_id field equals 1. This is a filtered query that only fetches comments associated with the note having note_id 1.

Query Purpose:

The purpose of this query is to identify and display comments linked to a specific note (in this case, the note with ID 1). It enables focused data retrieval for note-specific comment tracking or user feedback analysis.

Summary:

This query efficiently filters comments in the collection to those associated with a particular note. It is valuable for scenarios where comment data needs to be filtered by its relevance to specific notes, ensuring comprehensive note documentation and feedback tracking.



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.tasks.aggregate([ { $group:
 { _id: "$status", count: { $sum: 1 } } }] )
[ { _id: 'In Progress', count: 1 }, { _id: 'Completed', count: 1 } ]
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> _
```

Explanation of the Query

The query db.tasks.aggregate([{ $group: { _id: "$status", count: { $sum: 1 } } }]) uses MongoDB's aggregation framework to group documents in the tasks collection by their status field and count the number of tasks for each status.

Query Purpose:

The purpose of this query is to generate a summary report that provides the total count of tasks grouped by their status (e.g., "In Progress," "Completed"). This allows for a quick overview of task distribution based on their statuses.

Breakdown of the Aggregation Stages:

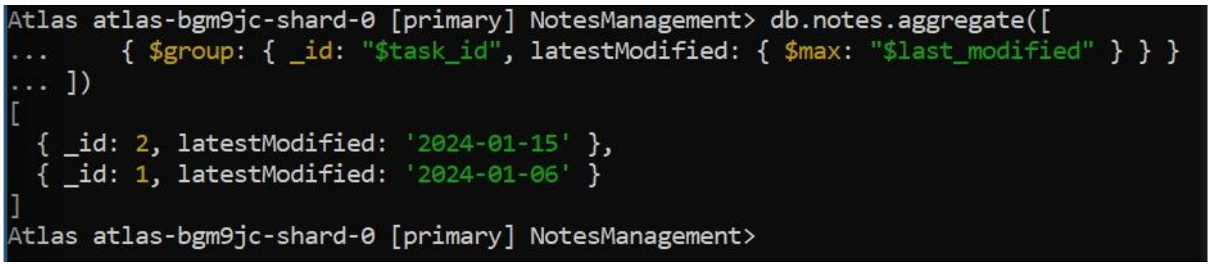
- **$group**:
  - _id: "$status": Groups the tasks based on the status field.
  - count: { $sum: 1 }: Sums up the total number of documents (tasks) in each group.

Example Output:

- { _id: 'In Progress', count: 1 }: Indicates that there is 1 task with the status "In Progress."
- { _id: 'Completed', count: 1 }: Indicates that there is 1 task with the status "Completed."

Summary:

This query provides a grouped count of tasks by their statuses, which is particularly useful for reporting and analytics. It helps identify the distribution of tasks in different statuses, aiding in project tracking and management insights.



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.notes.aggregate([
...     { $group: { _id: "$task_id", latestModified: { $max: "$last_modified" } } }
... ])
[
  { _id: 2, latestModified: '2024-01-15' },
  { _id: 1, latestModified: '2024-01-06' }
]
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement>
```

Explanation of the Query

The query db.notes.aggregate([{ $group: { _id: "$task_id", latestModified: { $max: "$last_modified" } } }]) uses MongoDB's aggregation framework to group documents in the notes collection by their task_id and find the latest modification date (last_modified) for each task.

Query Purpose:

The purpose of this query is to determine the most recent modification date for the notes associated with each task. This allows for tracking the latest updates made to the notes for specific tasks.

Breakdown of the Aggregation Stages:

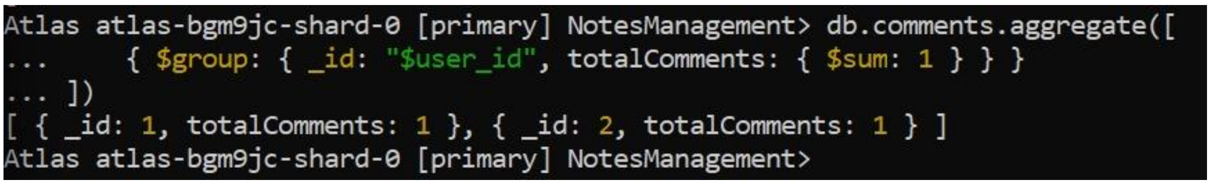
- **$group**:
  - _id: "$task_id": Groups the notes by their associated task_id.
  - latestModified: { $max: "$last_modified" }: Extracts the most recent modification date (last_modified) for each group.

Example Output:

- { _id: 2, latestModified: '2024-01-15' }: Indicates that the latest modification date for notes associated with task_id 2 is 2024-01-15.
- { _id: 1, latestModified: '2024-01-06' }: Indicates that the latest modification date for notes associated with task_id 1 is 2024-01-06.

Summary:

This query provides insights into the most recent updates for notes associated with each task, grouped by task_id. It is particularly useful for monitoring changes and ensuring that tasks are updated in a timely manner.



```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.comments.aggregate([
...     { $group: { _id: "$user_id", totalComments: { $sum: 1 } } }
... ])
[ { _id: 1, totalComments: 1 }, { _id: 2, totalComments: 1 } ]
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement>
```

Explanation of the Query

The query db.comments.aggregate([{ $group: { _id: "$user_id", totalComments: { $sum: 1 } } }]) uses MongoDB's aggregation framework to group documents in the comments collection by their user_id and count the total number of comments made by each user.

Query Purpose:

The purpose of this query is to calculate the total number of comments made by each user. It provides an aggregated view of user contributions in terms of comment activity.

Breakdown of the Aggregation Stages:

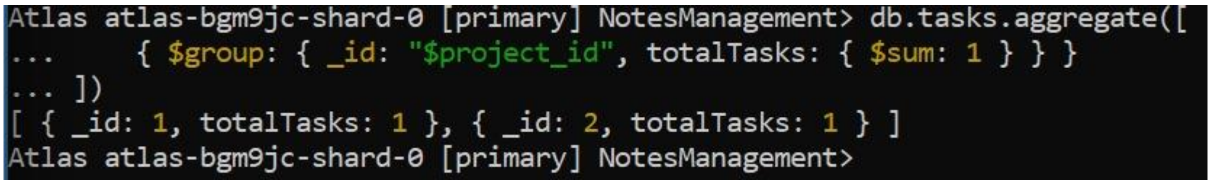
- **$group**:
    - _id: "$user_id": Groups the comments by the user_id field.
    - totalComments: { $sum: 1 }: Counts the total number of documents (comments) in each group by incrementing the sum by 1 for each document.

Example Output:

- { _id: 1, totalComments: 1 }: Indicates that the user with user_id 1 has made 1 comment.
- { _id: 2, totalComments: 1 }: Indicates that the user with user_id 2 has made 1 comment.

Summary:

This query provides a summary of the number of comments made by each user, grouped by their user_id. It is particularly useful for analyzing user engagement and contribution levels within the comment system.


```
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement> db.tasks.aggregate([
...     { $group: { _id: "$project_id", totalTasks: { $sum: 1 } } }
... ])
[ { _id: 1, totalTasks: 1 }, { _id: 2, totalTasks: 1 } ]
Atlas atlas-bgm9jc-shard-0 [primary] NotesManagement>
```

Explanation of the Query

The query db.tasks.aggregate([{ $group: { _id: "$project_id", totalTasks: { $sum: 1 } } }]) uses MongoDB's aggregation framework to group documents in the tasks collection by their project_id and count the total number of tasks associated with each project.

Query Purpose:

The purpose of this query is to calculate the total number of tasks associated with each project. It provides an aggregated view of task distribution across different projects.

Breakdown of the Aggregation Stages:

- **$group**:
  - _id: "$project_id": Groups the tasks by the project_id field.
  - totalTasks: { $sum: 1 }: Counts the total number of tasks in each group by incrementing the sum by 1 for each task.

Example Output:

- { _id: 1, totalTasks: 1 }: Indicates that project with project_id 1 has 1 associated task.
- { _id: 2, totalTasks: 1 }: Indicates that project with project_id 2 has 1 associated task.

Summary:

This query provides a summary of the number of tasks associated with each project, grouped by their project_id. It is particularly useful for project-level analysis and workload tracking within the task management system.