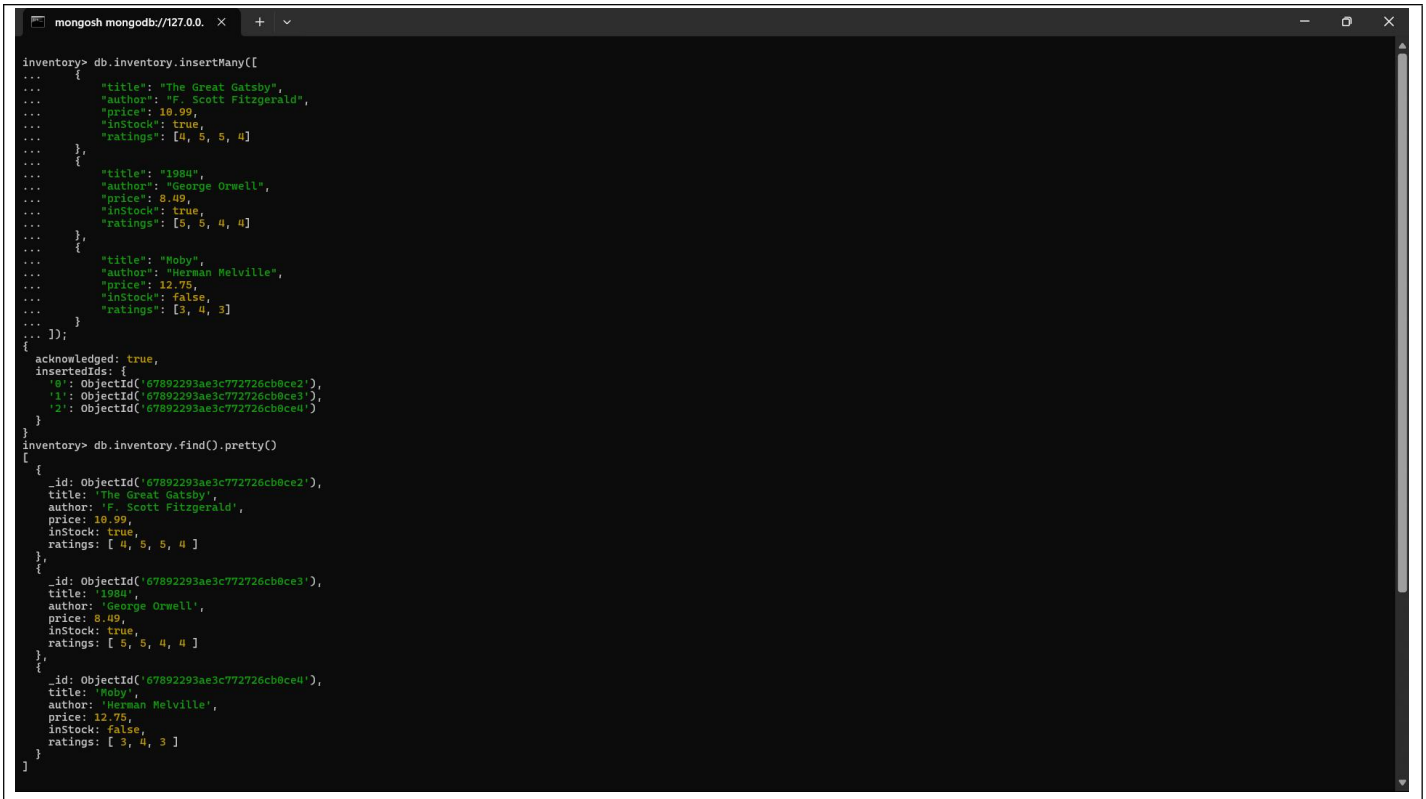
JEYSAN.V
717823F225
MERN TASK (mongo DB)

MongoDB Practice Exercises
1. Creating a New Database and Collection
Scenario:
You manage a small bookstore and need a database to track books.
Tasks:
Create a new database named bookstoreDB.
Within bookstoreDB, create a collection called inventory.
Ensure that bookstoreDB and inventory are visible in both MongoDB Compass and VS Code.
Document the commands (or steps in Compass) used to verify the database and collection creation.
Prepare to insert sample data into this new collection in the next exercise.

```
test> show dbs
admin   40.00 KiB
config  48.00 KiB
local   40.00 KiB
test> use bookstoreDB
switched to db bookstoreDB
bookstoreDB> db.createCollection("inventory")
{ ok: 1 }
bookstoreDB> show dbs
admin        40.00 KiB
bookstoreDB   8.00 KiB
config       72.00 KiB
local        40.00 KiB
bookstoreDB> use bookstoreDB
already on db bookstoreDB
bookstoreDB> use inventory
switched to db inventory
inventory> show collection
MongoshInvalidInputError: [COMMON-10001] 'collection' is not a valid argument for "show".
inventory> show collections
```

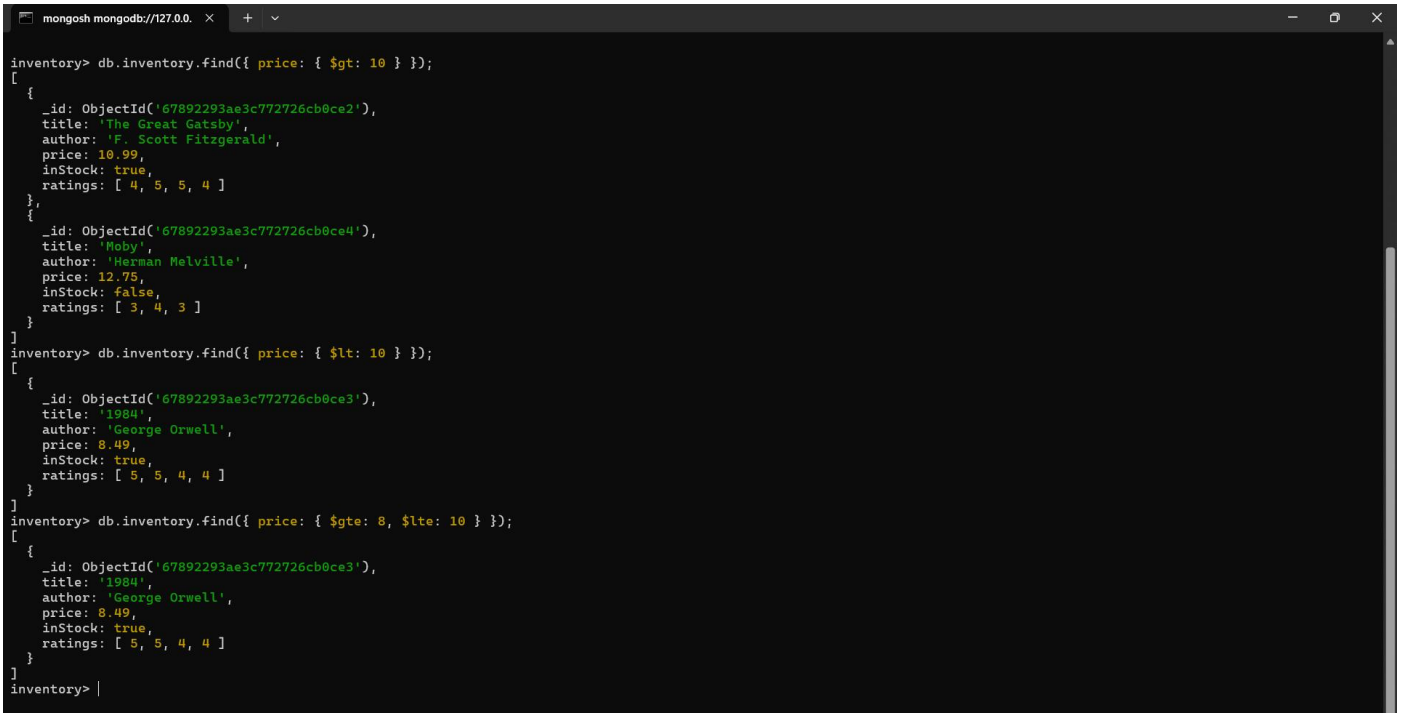
2. Inserting Multiple Documents
Scenario:
You have just received data for new books to add to your inventory.
Sample Data (JSON array):
[
{
"title": "The Great Gatsby",
"author": "F. Scott Fitzgerald",
"price": 10.99,
"inStock": true,
"ratings": [4, 5, 5, 4]
},
{
"title": "1984",
"author": "George Orwell",
"price": 8.49,
"inStock": true,
"ratings": [5, 5, 4, 4]
},
{
"title": "Moby",
"author": "Herman Melville",
"price": 12.75,
"inStock": false,
"ratings": [3, 4, 3]
}
]

Tasks:
Insert all these book documents into the inventory collection at once using insertMany.
Confirm the total number of documents in the collection.
Validate that each document’s fields match the provided sample data.
Record any errors or warnings you encounter.
Discuss how you would handle a scenario where a document is missing a required field (e.g., author).



3. Performing Basic find Queries
Scenario:
Customers often ask for specific books, and you want to query them by attributes like title or inStock.
Sample Data (already inserted from previous exercise).
Use the same documents:
"The Great Gatsby"
"1984"
"Moby"
Tasks:
Write a query to find all books currently in stock (inStock: true).
Search for a specific book by title (e.g., "Moby").
Experiment with returning only certain fields (e.g., title and price).
Verify the results in both Compass (using the Filter area) and VS Code (via scripts or commands).
Note how you would handle case-insensitive searches (e.g., searching great gatsby).

4. Utilizing Comparison Operators
Scenario:
You want to explore price-based filters, such as finding books below $10 or above $10.
Sample Data (from inventory collection):
"The Great Gatsby" (price: 10.99)
"1984" (price: 8.49)
"Moby" (price: 12.75)
Tasks:
Use $gt (greater than) to find books more expensive than $10.
Use $lt (less than) to find books cheaper than $10.
Experiment with $gte and $lte to search in or out of a specified price range (e.g., $8 to $10).
Compare results in Compass vs. a script in VS Code.
Note any differences in syntax or JSON usage when working with Compass vs. VS Code queries.

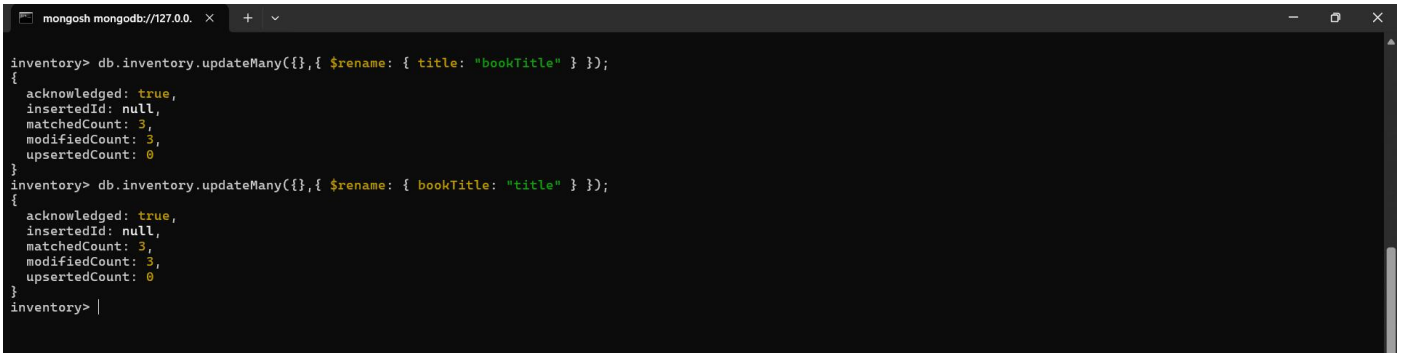


5. Updating Fields with $set and $unset
Scenario:
You discover that "1984" needs a price update, and you also want to remove a field that's no longer needed.
Sample Data (already existing in inventory):
"1984" (price: 8.49, inStock: true, ratings: [5, 5, 4, 4])
Tasks:
Increase "1984"'s price by setting it to a new value, for example, $9.49.
Remove (unset) a field called ratings from "1984" if you decide it's no longer needed.
Ensure the updates are reflected in both Compass and VS Code.
Observe how the document changes—especially noticing which fields remain after $unset.
Summarize how you might perform these updates on multiple documents at once.

6. Renaming Fields

MongoDB Practice Exercises

Scenario:
You realize the title field should be more explicit and decide to rename it to bookTitle.
Sample Data (already existing in inventory):
"The Great Gatsby"
"1984"
"Moby"
Tasks:
Rename title to bookTitle for all books in the collection using $rename.
Confirm the rename operation is successful and that title no longer appears.
Discuss potential impacts on any existing queries or client applications that expect title.
Document how you'd revert this change if needed.
Note how the rename appears in Compass and how you might verify it in a VS Code script.



7. Replacing an Entire Document
Scenario:
The information for "Moby" is inaccurate, so you want to replace its entire record.
Sample Data (new JSON):
{
"bookTitle": "Moby",
"author": "Herman Melville",
"price": 14.99,
"inStock": true,
"ratings": [4, 4, 3, 5],
"publisher": "Fiction Press"
}
Tasks:
Use the appropriate filter to find "Moby" by its current bookTitle value.
Replace the existing document with the new JSON above, removing all old fields.
Confirm that the old fields no longer appear in the updated document.

MongoDB Practice Exercises
Evaluate potential pitfalls if you replaced the document using the wrong filter.
Explore how partial updates differ from a complete replace in terms of data loss.

8. Building a Basic Aggregation Pipeline
Scenario:
You want to calculate the average price of your in-stock books.
Sample Data (already in inventory):
"The Great Gatsby" (price: 10.99)
"1984" (price: 9.49 or updated from previous step)
"Moby" (price: 14.99 or replaced)
Tasks:
Draft an aggregation pipeline to group all books and compute the average price.
In Compass, design the pipeline using the Aggregations panel.
In VS Code, outline the steps to execute the same pipeline with commands or code.
Make sure to exclude any books that are inStock: false from the average price calculation.
Record how you validate the output in both environments.

9. Combining $match and $project in Aggregations
Scenario:
You want a report showing only bookTitle and price for books priced at $10 or more.
Sample Data (from inventory):
"The Great Gatsby" (price: 10.99)
"1984" (price: ~9.49)
"Moby" (price: 14.99)
Tasks:
Create a $match stage to filter books with a price >= $10.
Use $project to show only bookTitle and price.
Exclude _id from the final output.
Verify the pipeline results in Compass and in VS Code.
Consider adding a sort stage to arrange books in ascending or descending order by price.

10. Deleting Documents and Dropping Collections
Scenario:
You're preparing to remove out-of-print books and possibly remove the entire inventory collection if it becomes unneeded.
Sample Data (existing inventory):
Any books you deem out-of-print or no longer available.
Tasks:
Select one book to delete by title (or bookTitle) with deleteOne.
Remove all books matching a certain criterion (e.g., inStock: false) with deleteMany.
Verify the remaining documents in Compass or by using a find query.
Practice dropping the entire inventory collection; note the steps for verification.
Reflect on how you would restore the dropped collection if it was deleted by mistake.