Ex.No.: 6

Import a JASON file from the command line. Apply the following actions with the data present in the JASON file where, projection, aggregation, remove, count, limit, skip and sort

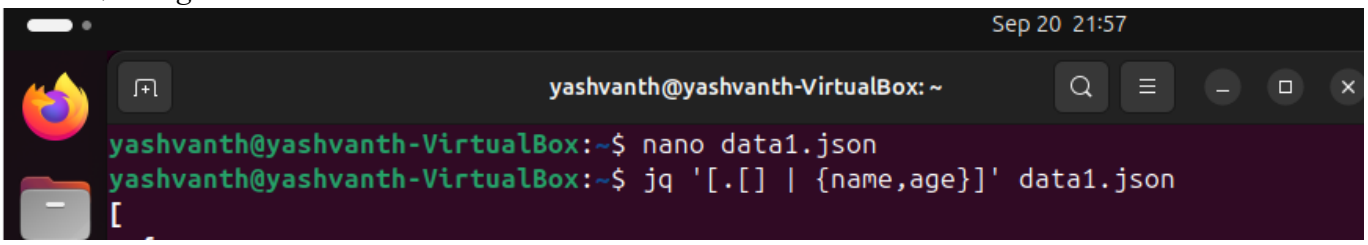
AIM:

To import a JASON file from the command line and apply the following actions with the data present in the JASON file where, projection, aggregation, remove, count, limit, skip and sort.

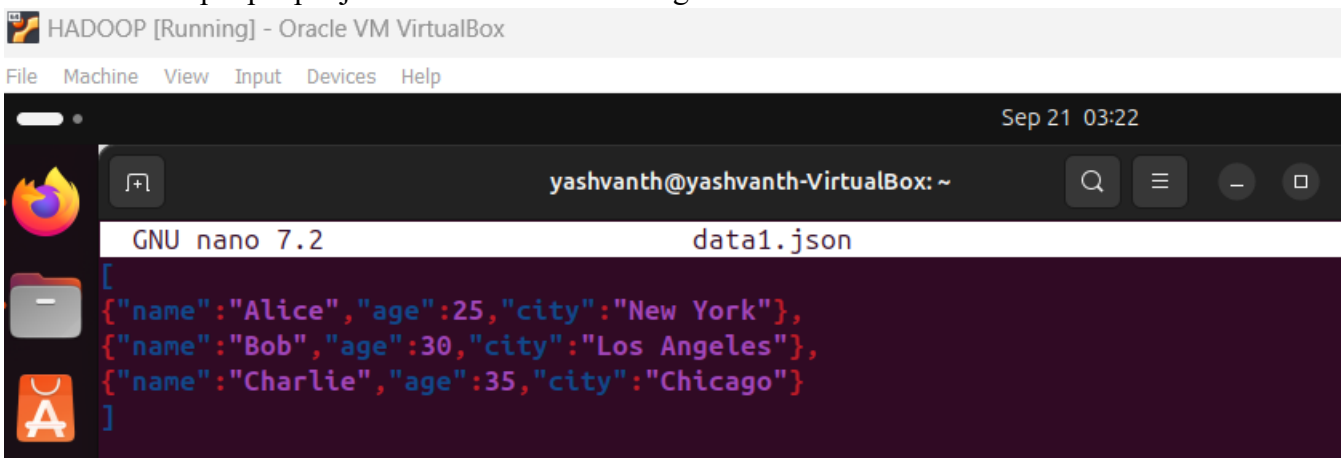
PROCEDURE:

1. Download and install MongoDB:

1. Visit the MongoDB Download Center and choose the version suitable for your operating system (Windows).
2. Select MSI as the package, and click "Download".
3. Run the downloaded MSI installer. Follow the prompts in the installation wizard.
4. Choose "Complete" for the setup type to install all components.
5. Select the option to install MongoDB as a service (recommended), which allows MongoDB to start automatically with your system.
6. Check the version installed by using the command
   $ mongod –version



2. Create a sample people.json file with the following content:



Import the JSON File

To import the JSON file into your MongoDB database, open your terminal or command prompt and use the mongoimport command:

>mongoimport --db mydb --collection people --file path_to_your_json/people.json –jsonArray

3. Download and install MongoShell:

1. Visit the MongoDB Shell https://www.mongodb.com/try/download/shell Download Page.
2. Select your operating system as Windows and download the MSI package. Run the downloaded MSI installer.
3. Follow the installation prompts to complete the installation.
4. Ensure the option to add MongoDB Shell to your PATH is checked during installation.
5. To verify, open Command Prompt or PowerShell and type the following command to check if mongosh is installed correctly:

   $ mongosh --version
6. If installed correctly, it will display the version of mongosh.
7. To start MongoDB Shell open Command Prompt Type mongosh to start the MongoDB Shell. It will connect to the default MongoDB server (localhost:27017).

4. Create and Switch to the Database:

Use the use command to create and switch to the new database. Then, switch to your database and check the collection:

test> use mydb
mydb> show collections

>db.people.find().pretty()

```
yashvanth@yashvanth-VirtualBox:~$ jq 'sort_by(.age)' data1.json
[
  {
    "name": "Alice",
    "age": 25,
    "city": "New York"
  },
  {
    "name": "Bob",
    "age": 30,
    "city": "Los Angeles"
  },
  {
    "name": "Charlie",
    "age": 35,
    "city": "Chicago"
  }
]
```

Step 4: Perform Actions on the JSON Data

Now, we will apply the following operations: where, projection, aggregation, remove, count, limit, skip, and sort.

1. **Length:** Find records where the city is "New York":
   >db.people.length})

```
yashvanth@yashvanth-VirtualBox:~$ jq 'length' data1.json
3
```

2. **Remove:** Delete people who live in "Chicago":
   >db.data.deleteMany({ city: "Chicago" })

```
yashvanth@yashvanth-VirtualBox:~$ jq 'sort_by(.age)' data1.json
[
  {
    "name": "Alice",
    "age": 25,
    "city": "New York"
  },
  {
    "name": "Bob",
    "age": 30,
    "city": "Los Angeles"
  },
  {
    "name": "Charlie",
    "age": 35,
    "city": "Chicago"
  }
]
```

3. **Count:** Count the total number of people in the collection:
   >db.data.count

```
yashvanth@yashvanth-VirtualBox:~$ jq '.[2:]' data1.json
[
  {
    "name": "Charlie",
    "age": 35,
    "city": "Chicago"
  }
]
```

4. **Skip:** Skip the first record and show the rest:
   >db.data.skip

```
yashvanth@yashvanth-VirtualBox:~$ jq '[.[] | {name,age}]' data1.json
[
  {
    "name": "Alice",
    "age": 25
  },
  {
    "name": "Bob",
    "age": 30
  },
  {
    "name": "Charlie",
    "age": 35
  }
]
```

Trash

5. **Sort:** Sort the records by age in ascending order:
   >db.data.find().sort

```
data.json
[
  {
    "name": "Bob",
    "age": 25
  },
  {
    "name": "Alice",
    "age": 30
  }
]
```

RESULT:

Thus to import a JASON file from the command line and apply the following actions with the data present in the JASON file where, projection, aggregation, remove, count, limit, skip and sort has been executed and verified successfully.