|  |  |  |
| --- | --- | --- |
| **UMBC – CSEE Department** |  | Dr. Waleed Youssef |
| **Data Science Program** |  | youssef1@umbc.edu |
| **Spring 2023** |  |  |
| **DATA 603 – Big Data Platforms** | | |
| **Homework #4** | | |
|  |  |  |

**Questions:**

1. **[10 points]** Describe when NoSQL database is preferred to RDBMS? Present benefits and different scenarios.

**Flexible Data Requirements:**

NoSQL has the ability to handle flexible data storage. Traditional RDMS has static data structure because changing structure is often a time-consuming and expensive. Whereas NoSQL can handle an ever-changing data model. For scenarios like geo-spatial data, engineering parts, others data constantly changes and evolving.

**Rapid Data Growth and Scalability:**

RDBMS scales vertically i.e. when the database faces performance issues it migrates to larger hard drive or faster server. Where as NoSQL database scales horizontally, i.e., adds more servers so that load is distributed across multiple servers. At scenarios involving the launch of a new app or service, if a company’s customer base can suddenly skyrocket into millions, NoSQL can handle it perfectly by scaling resources faster.

**Concurrent Performance:**

When there are high volume of concurrent users NoSQL is best. Most relational databases give priority to data integrity which uses locking mechanism i.e., when a transaction starts there is a lock on it so that others process can’t modify it. But it impacts the performance when the number of users increases. NoSQL has better mechanism to handle it. [1]

1. **[10 points]** Compare MongoDB to Cassandra DB, discuss advantages and disadvantages?

**Cassandra:**

Cassandra is an open-source project. It provides modern applications with continuous availability without downtime. Cassandra provides seamless replication facility across data centers and zones. It can handle petabytes of data and perform multiple concurrent operations within seconds.

**MongoDB:**

MongoDB is an open-source project which is a versatile and scalable NoSQL document database platform. It is popular for horizontal scaling and load balancing features.

**Similarities**:

* Both are open-source NoSQL distributed databases.
* Both can horizontally scale.
* Both databases support horizontal partitioning and replication.

**Differences:**

* MongoDB is a document-based database with collections containing multiple documents, where as Cassandra is a column-oriented database.
* MongoDB has master-slave architecture, while Cassandra has a peer-to-peer architecture where all are master nodes in communication with each other.
* Cassandra has no single-point failure, while MongoDB can have single-point failure with its master, but this could be repaired quickly by switching the master.
* MongoDB fulfils consistency and partitioning tolerance, whereas Cassandra is highly available with partition tolerance. MongoDB sacrifice availability, whereas Cassandra gives up consistency.
* MongoDB uses JSON formats to store documents, while Cassandra uses a columnar style and tables.

**Advantages of Cassandra:**

* It is an open-source technology with peer-to-peer architecture which eliminates a single point of failure
* Cassandra is highly scalable.
* It supports data replication hence it is fault-tolerant and has high availability.
* It can easily handle massive amounts of data, and writes extremely fast.

**Disadvantages of Cassandra:**

* It doesn’t support ACID properties
* It does not support aggregates.
* It is optimized for fast writes hence reads slow.

**Advantages of MongoDB:**

* Open-source, scalable NoSQL database.
* It is schema-less database that supports aggregation.
* Both community and enterprise versions are available.
* Consistency and availability are inherent.

**Disadvantages of MongoDB:**

* Complex joins are not possible.
* High memory usage.
* Limited nesting and document data size. [2]

1. **[15 points]** Describe new features of MongoDB not shared in class, especially features available in releases that were not part of older releases

**Quiesce Period:** Starting from MongoDB 5.0 mongod enter a quiesce period to allow any ongoing database operations to complete before shutting down.

**Resharding**

With sharding, the cluster of database can share larger datasets and handle additional developer requests.

The ideal shard key allows MongoDB to distribute documents evenly throughout the cluster while facilitating common query patterns. A suboptimal shard key can lead to performance or scaling issues due to uneven data distribution. Starting in MongoDB 5.0, you can use the [reshardCollection](https://www.mongodb.com/docs/manual/reference/command/reshardCollection/" \l "mongodb-dbcommand-dbcmd.reshardCollection) command to change the shard key for a collection to change the distribution of your data across your cluster.

**Time Series Collections**

MongoDB 5.0 introduces time series collections which efficiently store sequences of measurements over a period of time. Compared to normal collections, storing time series data in time series collections improves query efficiency and reduces disk usage for your data and indexes. [3]

**Authentication:**

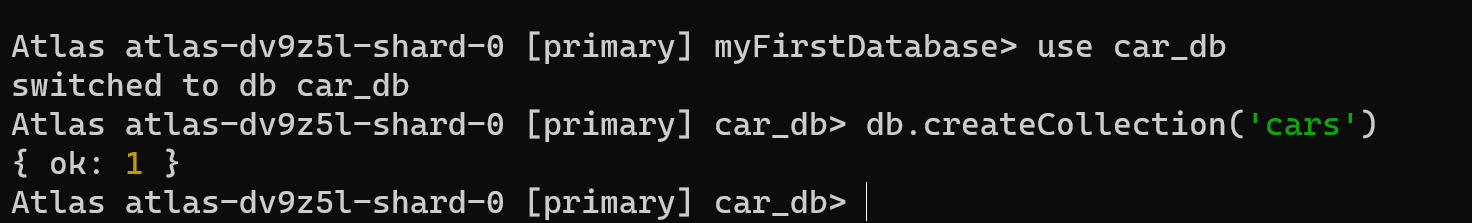
It ensures that only authorized users can access database. Without authentication, anyone can access your data.

MongoDB provides a number of authentication mechanisms to access database. The most common is the Salted Challenge Response Authentication Mechanism, which is the default one. SCRAM required users to username and password to used database. [4]

1. **[25 points]** MongoDB Implementation Assignment:
2. Create and develop a new MongoDB database for car makes and models.

Created a database car\_db using “**use car\_db**”

Created a collection cars using “**db.createCollection('cars')**”



1. Generate one report of data loaded, showing different commands applied on this database (load at least 20 records in the database).

db.cars.insertMany([

{make:'Jeep',model:'Grand Cherokee',country:'US'},

{make:'Jeep',model:'Grand Cherokee 4xe',country:'US'},

{make:'Jeep',model:'Wrangler 4xe',country:'US'},

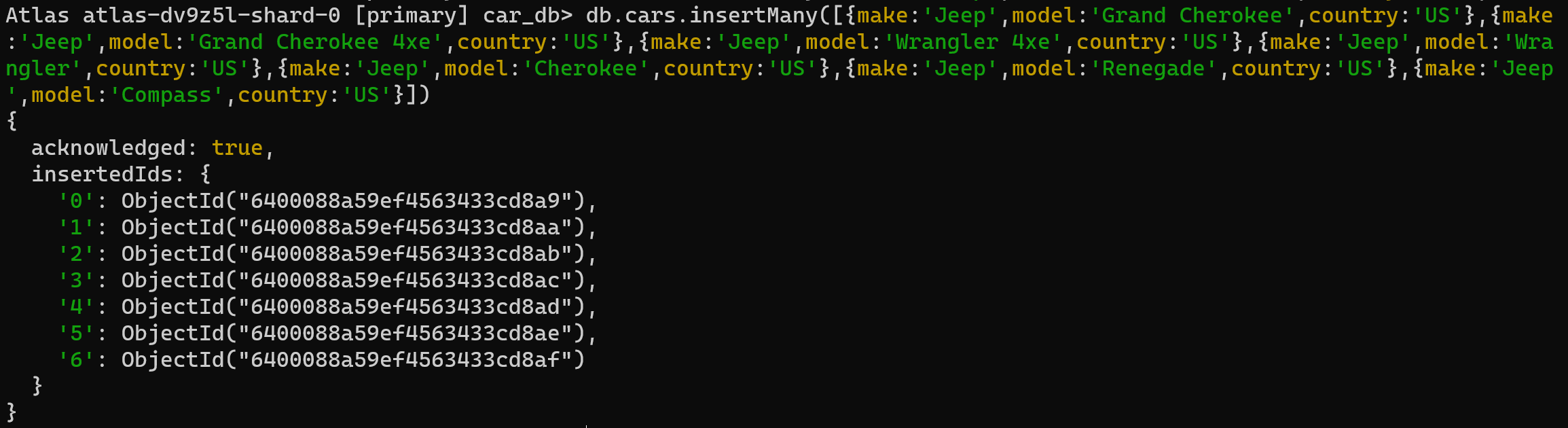
{make:'Jeep',model:'Wrangler',country:'US'},

{make:'Jeep',model:'Cherokee',country:'US'},

{make:'Jeep',model:'Renegade',country:'US'},

{make:'Jeep',model:'Compass',country:'US'}

])



db.cars.insertMany([{make:'Tata',model:'Tiago',country:'INDIA'},{make:'Tata',model:'Altroz',country:'INDIA'},{make:'Tata',model:'Tigor',country:'INDIA'},{make:'Tata',model:'Punch',country:'INDIA'},{make:'Tata',model:'Nexon',country:'INDIA'},{make:'Tata',model:'Harrier',country:'INDIA'},{make:'Tata',model:'Safari',country:'INDIA'},{make:'Tata',model:'Nexon EV',country:'INDIA'}])

db.cars.insertMany([{make:'Tesla',model:'Model S',country:'USA'}, {make:'Tesla',model:'Model 3',country:'USA'},{make:'Tesla',model:'Model X',country:'USA'},{make:'Tesla',model:'Model Y',country:'USA'}])

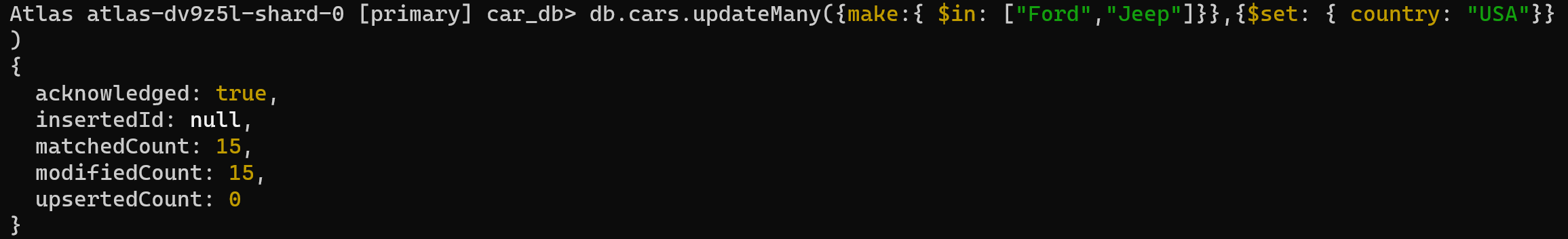
db.cars.insertMany([{make:'BMW',model:'F40',country:'Germany'},{make:'BMW',model:'X1',country:'Germany'},{make:'BMW',model:'X7',country:'Germany'},{make:'BMW',model:'2 Series Active Tourer',country:'Germany'},{make:'BMW',model:'i3 (G28)',country:'Germany'},{make:'BMW',model:'iX1',country:'Germany'},{make:'BMW',model:'iX3',country:'Germany'}])

db.cars.insertMany([{make:'Kia',model:'EV6',country:'South Korea'},{make:'Kia',model:'Niro',country:'South Korea'},{make:'Kia',model:'Niro EV',country:'South Korea'},{make:'Kia',model:'Niro Plug-In Hybrid',country:'South Korea'},{make:'Kia',model:'Seltos',country:'South Korea'},{make:'Kia',model:'Sorento',country:'South Korea'}])

db.cars.insertMany([{make:'Ford',model:'Bronco',country:'US'},{make:'Ford',model:'Bronco Sport',country:'US'} ,{make:'Ford',model:'EcoSport',country:'US'}, {make:'Ford',model:'Edge',country:'US'},{make:'Ford',model:'Escape',country:'US'},{make:'Ford',model:'Escape PHEV',country:'US'},{make:'Ford',model:'Expedition',country:'US'} ,{make:'Ford',model:'Expedition Max',country:'US'}])

updating country to USA for brands Ford and Jeep as it is US

db.cars.updateMany({make:{ $in: ["Ford","Jeep"]}},{$set: { country: "USA"}})



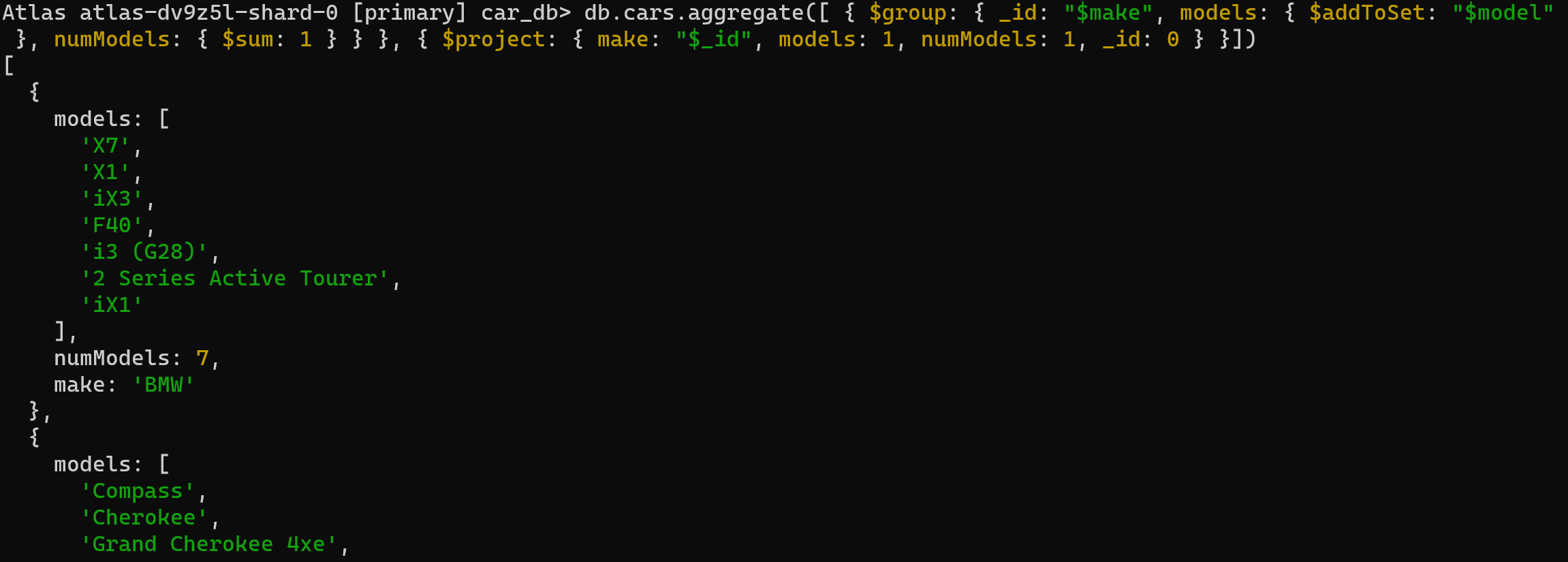
Below document shows all records in the collection cars



1. Generate one report showing how many models you entered per car make.

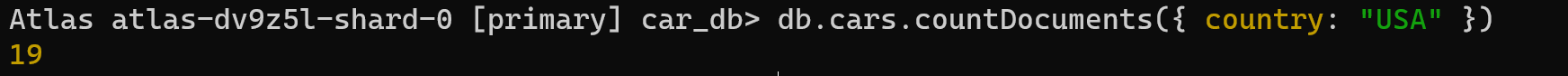
db.cars.aggregate([ { $group: { \_id: "$make", models: { $addToSet: "$model" }, numModels: { $sum: 1 } } }, { $project: { make: "$\_id", models: 1, numModels: 1, \_id: 0 } }])



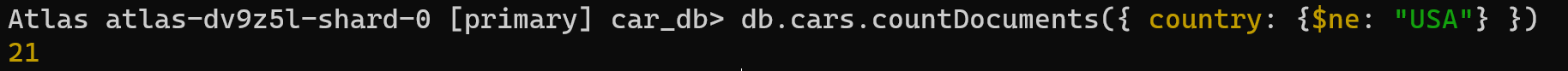


1. Generate another report showing how many American-made cars versus rest of the world.

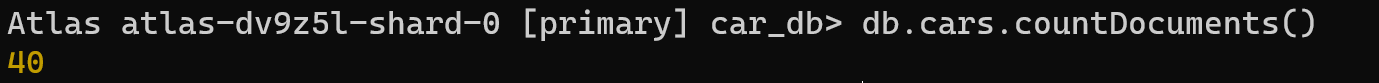
db.cars.countDocuments({ country: "USA" })



db.cars.countDocuments({ country: {$ne: "USA"} })



Total documents:



Is equal to sum of USA based models and non USA based models

* + ***Hints:***
    - *In your report, you should list all commends used starting from creating the database, creating collections and documents, inserting data, and querying/filtering data*
    - *Show your work to get full credit*

**References:**

[1] Murray, A. (2021, February 17). When to Consider a NoSQL vs Relational Database | Mend. from <https://www.mend.io/resources/blog/when-to-consider-a-nosql-vs-relational-database/>

[2] Madhugiri, D. (2023, February 12). *Cassandra vs MongoDB : Key Differences You Should Know!* KnowledgeHut. from <https://www.knowledgehut.com/blog/data-science/cassandra-vs-mongodb>

[3] *Release Notes for MongoDB 5.0 — MongoDB Manual*. (n.d.). MongoDB, from <https://www.mongodb.com/docs/manual/release-notes/5.0/>

[4] *MongoDB Features & Key Characteristics | MongoDB*. (n.d.). MongoDB, from <https://www.mongodb.com/features>