Introduction to MongoDB

Sophia Sandhu

Agenda

- Mongo Db History
- MongoDb Characteristics
- MongoDB Objects
- Replication
- Sharding
- MongoDb Lab

MongoDB's History

- Developed by 10gen
- Founded in 2007
- A document-oriented, NoSQL database
- Hash-based, schema-less database
  - No Data Definition Language - you can store hashes with any keys and values that you choose
  - Keys are stored as strings
  - Document Identifiers (_id) will be created for each document, field name reserved by system.
  - Uses BSON format
- Based on JSON
- Written in C++
- Supports APIs (drivers) in many computer languages like : JavaScript, Python, Ruby, Perl, Java, C#, C++

What is MongoDB?

- Document-type NoSQL database
- Open source (MongoDB Inc.)
- Multi-platform
- Documents in JSON-like format
- JavaScript is integrated into the environment
- Flexible schema
- Editions:
  - MongoDB Community Server (Free)
  - MongoDB Enterprise Server (Commercial edition)
  - MongoDB Atlas (Cloud service)
- Mongo shell used to manage databases
- MongoDB Compass provides GUI-based database management

MongoDB Characteristics

- Dynamic schema
  - No DDL
- Document-based database.
- Query language via an API
- Atomic writes and fully-consistent reads.
- Support for embedded data models reduces I/O activity on database system.
- Indexes support faster queries and can include keys from embedded documents and arrays
- Master-slave replication with automated failover (replica sets)
- Automated range-based partitioning of data (sharding).
- Focuses on consistency and partition tolerance (CAP Theorem)

MongoDB Characteristics

- MongoDB's replication facility, called replica set, provides:
  - *automatic* failover
  - data redundancy.

- A replica set is a group of MongoDB servers that maintain the same data set, providing redundancy and increasing data availability.

- MongoDB provides horizontal scalability as part of its *core* functionality : Sharding distributes data across a cluster of machines.

MongoDB Objects

- Database – 0 or more databases
  - Collections - 1 or more Collections
    - Documents - 0 or more documents
      - Fields- 1 or more fields

RDBMS to NoSql

RDBMS		MongoDB
Database		Database
Table	→	Collection
Row	→	Document
Columns	→	Field
Index	→	Index
Join	→	Embedded document
Partition	→	Shard

Schema Free Documents

- Unlike SQL databases, where you must determine and declare a table's schema before inserting data, MongoDB's collections, by default, do not require their documents to have the same schema. That is:
  - MongoDB does not need any pre-defined data schema.
  - Every document in a collection could have different data.
- A record in MongoDB is a document, which is a data structure composed of field and value pairs.
- MongoDB documents are similar to JSON objects. The values of fields may include other documents, arrays, and arrays of documents.

```
{
  "_id": {
        "$oid": "606ddf95a850f903fc25fce6"    },
   "FirstName": "Sophia",
   "LastName": "Sandhu",
   “major": "Computer Science",
   "phone":
          ["647-303-4567",
           "234-567-6789"],
   "gpa": 3.9
}
```

Schema Free Documents

Document 1

```
{
  "_id": {      "$oid": "606ddf95a850f903fc25fce6"    },
"FirstName": "Sophia",
 "LastName": "Sandhu",
  "major": "Computer Science",
  "phone": [
                "647-303-4567",
                "234-567-6789"],
 "gpa": 3.5
}
```

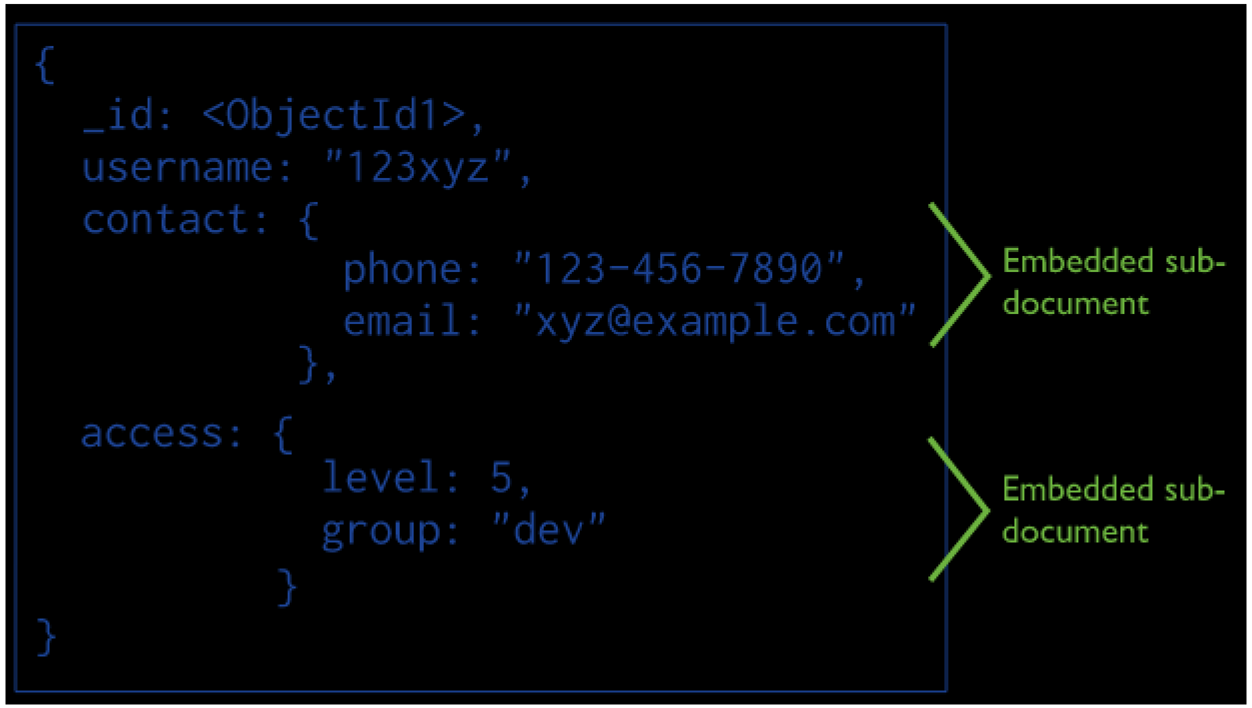
Document 2

```
{
 "_id": {      "$oid": "606de16349f3bd3718aa7361"    },
“StudentNo": "9900001",
 "FirstName": "John",
 "LastName": "Doe",
  "Grade": 85,
  "StudentType": "Domestic",
   "major": "Electrical“
}
```

- MongoDB does not need any pre-defined data schema.
- Every document in a collection could have different data.

Embedded Documents

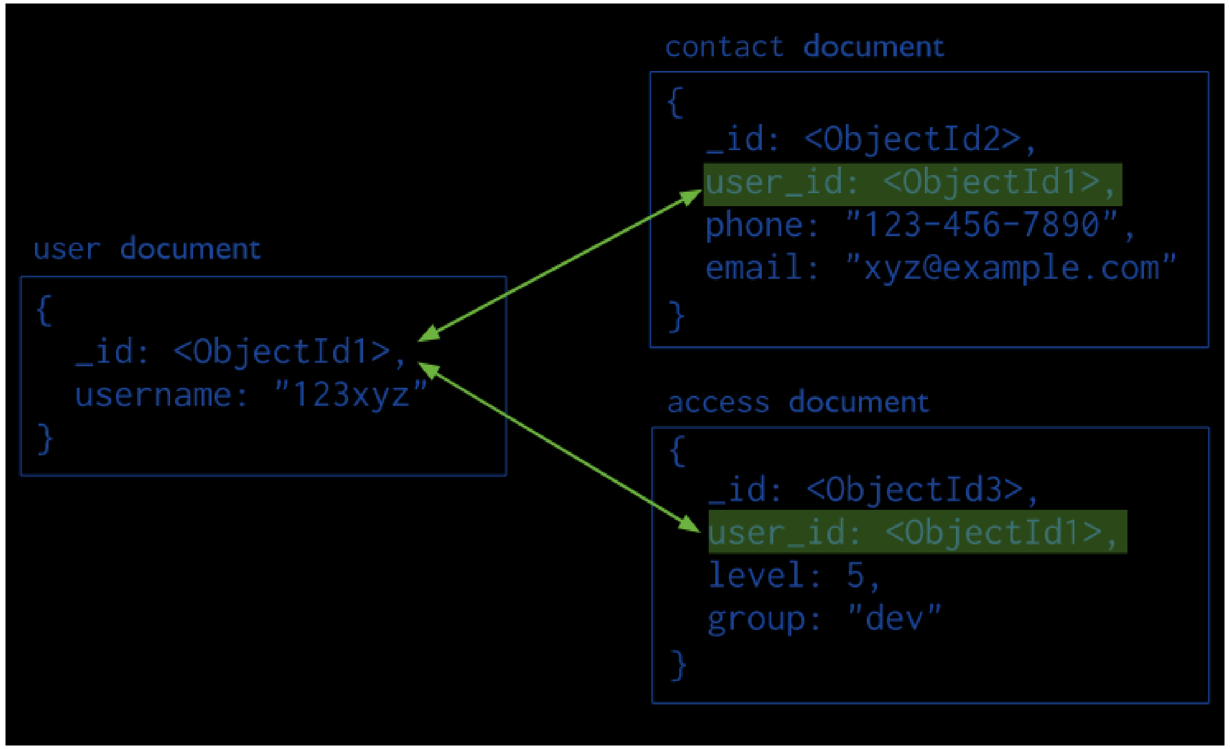
- Embedded documents capture relationships between data by storing related data in a single document structure.
- MongoDB documents make it possible to embed document structures in a field or array within a document.



```
{
  _id: <ObjectId1>,
  username: "123xyz",
  contact: {
              phone: "123-456-7890",
              email: "xyz@example.com"
            },
  access: {
             level: 5,
             group: "dev"
           }
}
```

References

- References store the relationships between data by including links or references from one document to another. Applications can resolve these references to access the related data.
- Data model using references to link documents.
- Both the ``contact`` document and the ``access`` document contain a reference to the ``user`` document.



user document

```
{
  _id: <ObjectId1>,
  username: "123xyz"
}
```

contact document

```
{
  _id: <ObjectId2>,
  user_id: <ObjectId1>,
  phone: "123-456-7890",
  email: "xyz@example.com"
}
```

access document

```
{
  _id: <ObjectId3>,
  user_id: <ObjectId1>,
  level: 5,
  group: "dev"
}
```

Index

- B+ tree indexes
- An index is automatically created on the _id field (the primary key)

- Users can create other indexes to improve query performance or to enforce Unique values for a particular field.
  - Supports single field index as well as Compound index
  - Like SQL order of the fields in a compound index matters
  - If you index a field that holds an array value, MongoDB creates separate index entries for every element of the array

- Sparse property of an index ensures that the index only contain entries for documents that have the indexed field. (so ignore records that do not have the field defined)

- If an index is both unique and sparse – then the system will reject records that have a duplicate key value but allow records that do not have the indexed field defined

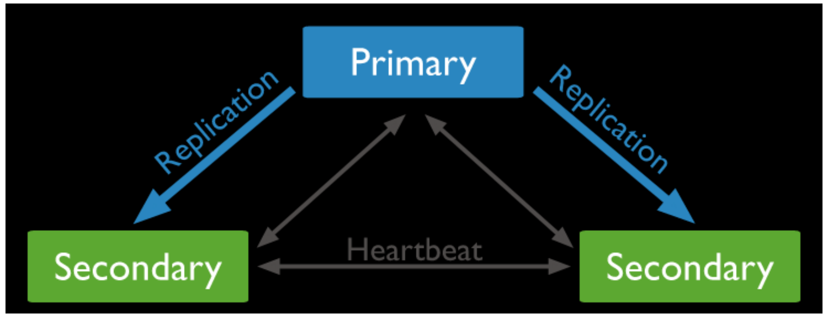
Replication

mongoDB

- A replica set in MongoDB is a group of mongod processes that maintain the same data set.
- Replica sets provide redundancy and high availability, and are the basis for all production deployments.
- With multiple copies of data on different database servers, replication provides a level of fault tolerance against the loss of a single database server.
- Replication can provide increased read capacity as clients can send read operations to different servers. Maintaining copies of data in different data centers can increase data locality and availability for distributed applications

MongoDB : Replication

- A replica set is a group of mongod instances that maintain the same data set.
- A replica set contains several data bearing nodes and optionally one arbiter node. Of the data bearing nodes, one and only one member is deemed the primary node, while the other nodes are deemed secondary nodes.
- The primary node receives all write operations. The primary records all changes to its data sets in its operation log, i.e. oplog.
- The secondaries replicate the primary's oplog and apply the operations to their data sets such that the secondaries' data sets reflect the primary's data set.
- If the primary is unavailable, an eligible secondary will hold an election to elect itself the new primary.



MongoDB : Replication

- When a primary does not communicate with the other members of the set for more than the configured electionTimeoutMillis period (10 seconds by default), an eligible secondary calls for an election to nominate itself as the new primary.

- The cluster attempts to complete the election of a new primary and resume normal operations.

- By default, clients read from the primary however, clients can specify a read preference to send read operations to secondary.

Sharding

- Sharding is a method for distributing data across multiple machines. MongoDB uses sharding to support deployments with very large data sets and high throughput operations.

- Database systems with large data sets or high throughput applications can challenge the capacity of a single server. For example, high query rates can exhaust the CPU capacity of the server. Working set sizes larger than the system's RAM stress the I/O capacity of disk drives.

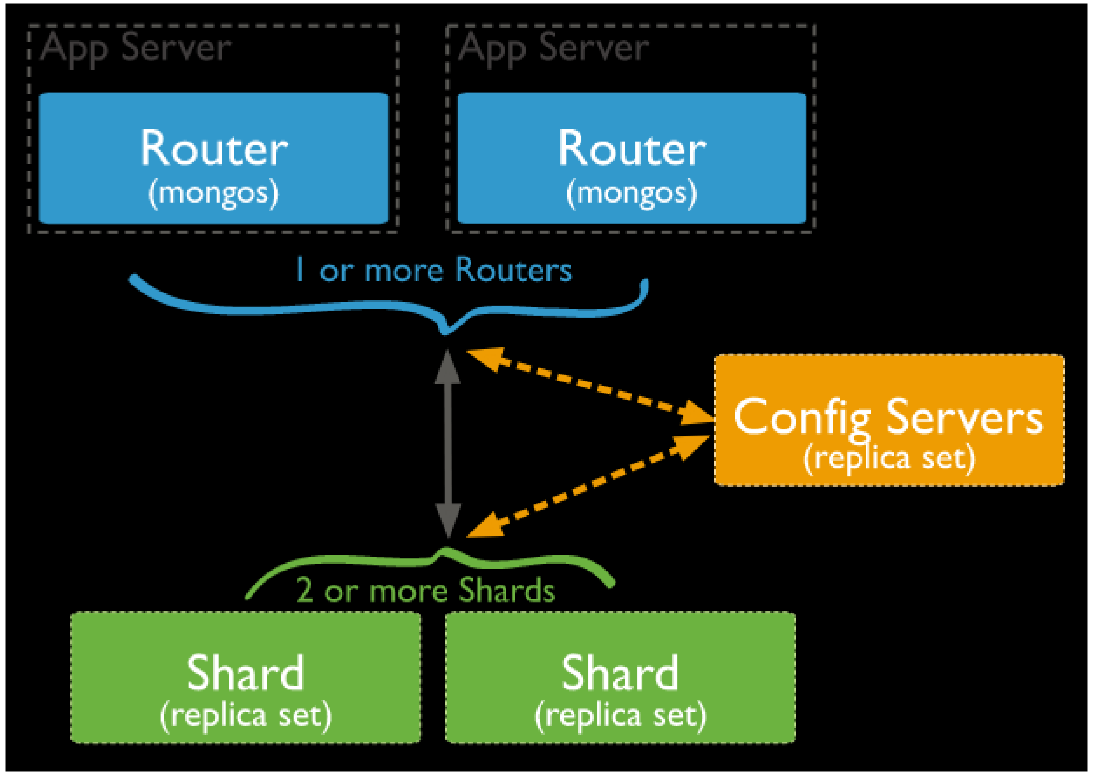
- There are two methods for addressing system growth: vertical and horizontal scaling.

  - **Vertical Scaling** involves increasing the capacity of a single server, such as using a more powerful CPU, adding more RAM, or increasing the amount of storage space.
    - Limitations in available technology may restrict a single machine from being sufficiently powerful for a given workload

  - **Horizontal Scaling** involves dividing the system dataset and load over multiple servers, adding additional servers to increase capacity as required. While the overall speed or capacity of a single machine may not be high, each machine handles a subset of the overall workload, potentially providing better efficiency than a single high-speed high-capacity server.

MongoDB: Sharding

- Mongo DB supports horizontal scaling through Sharding
- A MongoDB sharded cluster consists of the following components:
  - **Shard** : Each shard contains a subset of the sharded data. Each shard can be deployed as a replica set.
  - **mongos:** The mongos acts as a query router, providing an interface between client applications and the sharded cluster.
  - **config servers:** Config servers store metadata and configuration settings for the cluster.



MongoDB: Sharding

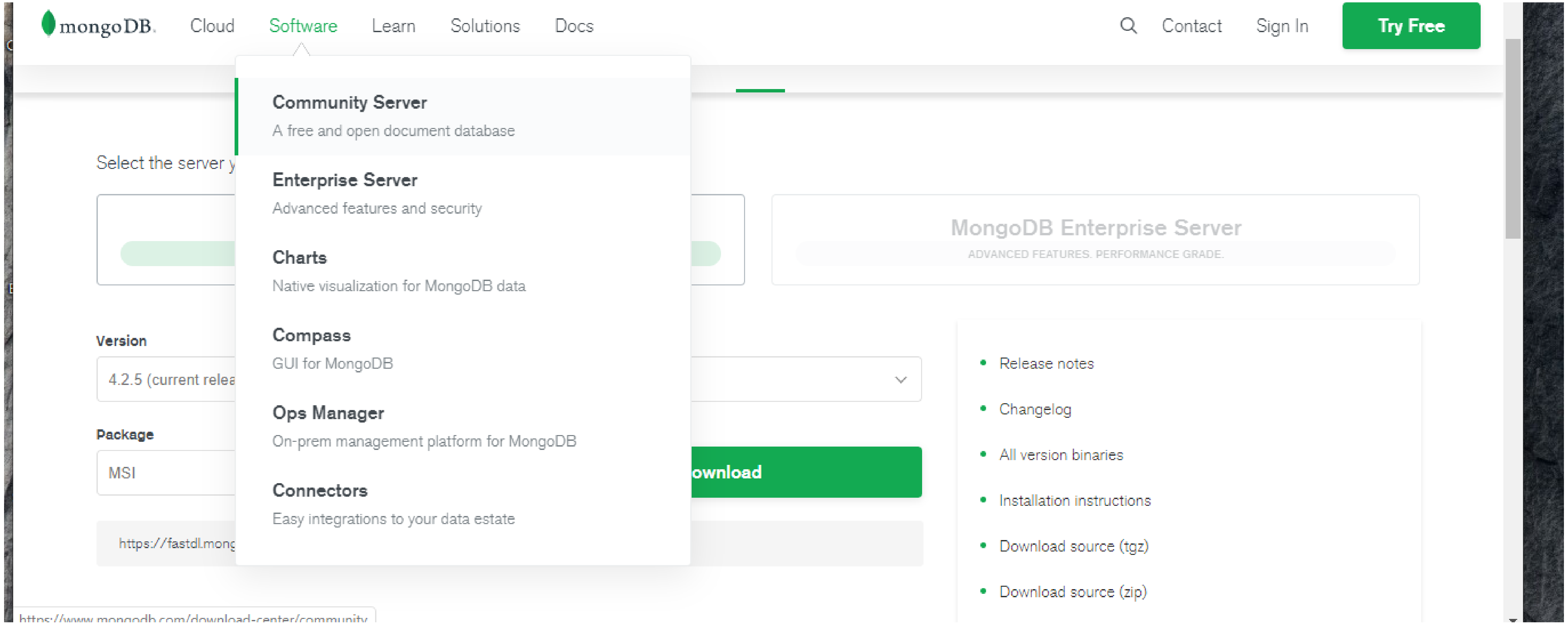
- MongoDB uses the shard keys to distribute the collection's documents across shards. The shard key consists of a field or multiple fields in the documents.
- A document's shard key value determines its distribution across the shards.
- To shard a populated collection, the collection must have an index that starts with the shard key.
- MongoDB partitions sharded data into chunks . Each chunk has an inclusive lower and exclusive upper range based on the shard key.
- MongoDB supports two sharding strategies for distributing data across sharded clusters:
  - Hashed Sharding
  - Ranged Sharding

MONGODB LAB

Installing MongoDB: Server

- Go to the MongoDB web site and download appropriate server (Community Server) in: https://www.mongodb.com/download-center/community
- Current version is 4.5.5
- Ensure the bin folder for Mongo is in the path to allow access in any directory
  - In bin folder (for Windows):
  - mongo.exe is the mongo shell
  - mongod.exe is the daemon
  - Similar executable file names are used for Linux and MacOS

Installing MongoDB



Community Server
A free and open document database

Enterprise Server
Advanced features and security

Charts
Native visualization for MongoDB data

Compass
GUI for MongoDB

Ops Manager
On-prem management platform for MongoDB

Connectors
Easy integrations to your data estate

Select the server y

MongoDB Enterprise Server
ADVANCED FEATURES. PERFORMANCE GRADE.

Version
4.2.5 (current relea

Package
MSI

ownload

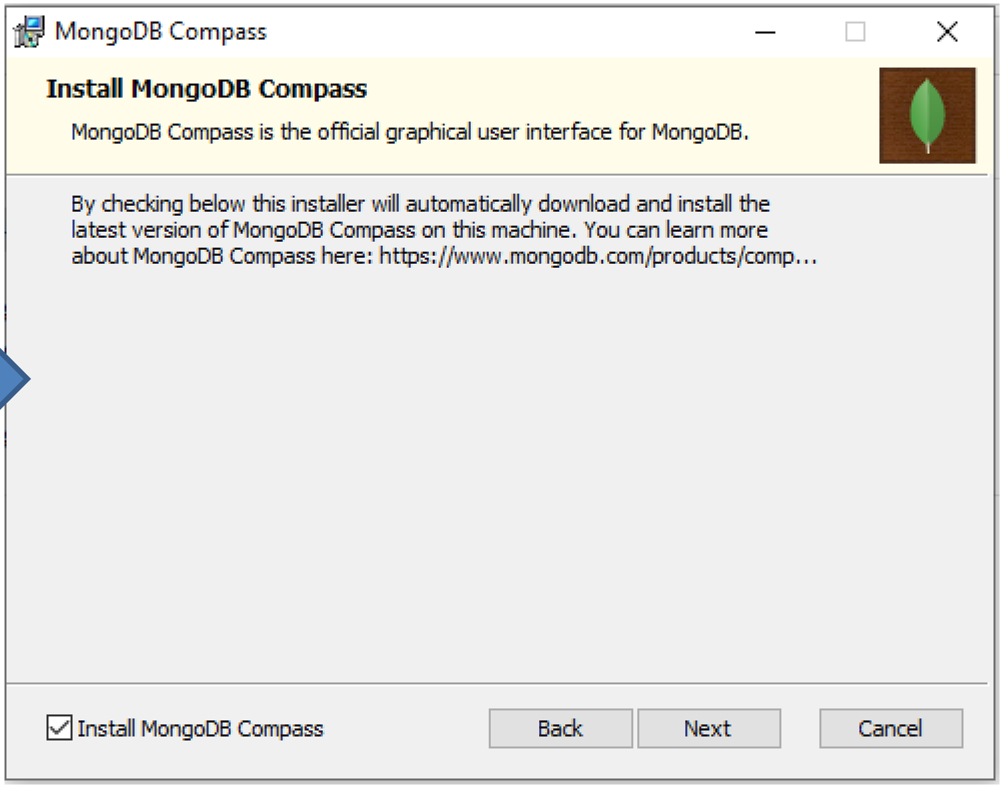
https://fastdl.mong

- Release notes
- Changelog
- All version binaries
- Installation instructions
- Download source (tgz)
- Download source (zip)

https://www.mongodb.com/download-center/community

Installing MongoDB

- Once download is complete open the msi file. Click Next in the start up screen.
- Accept the End-User License Agreement
- Click Next
- When you are on choose Setup type screen, make sure to select "Complete" option.
- Click next



Install MongoDB Compass

MongoDB Compass is the official graphical user interface for MongoDB.

By checking below this installer will automatically download and install the latest version of MongoDB Compass on this machine. You can learn more about MongoDB Compass here: https://www.mongodb.com/products/comp...

☑ Install MongoDB Compass

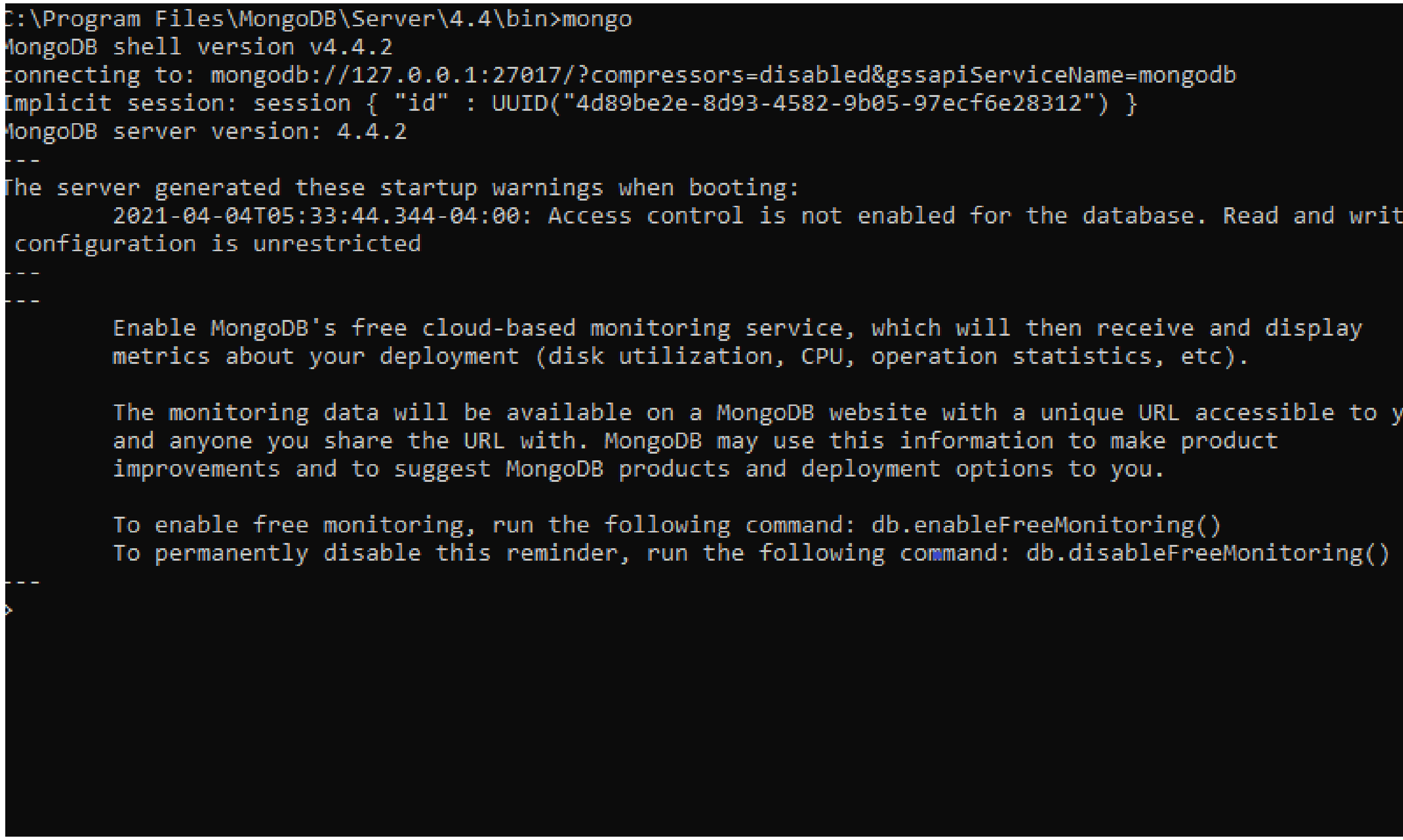
Installing MongoDB

- Click on the Install button to start the installation.
- Follow the installation steps and click on “finish” button to complete the installation.

Using Mongo Shell

- Open the command prompt and goto the MongoDB Server->Server-> bin folder
- Enter mongo

```
cd C:\Program Files\MongoDB\Server\4.4\bin
C:\Program Files\MongoDB\
   Server\4.4\bin>mongo
```



```
C:\Program Files\MongoDB\Server\4.4\bin>mongo
MongoDB shell version v4.4.2
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("4d89be2e-8d93-4582-9b05-97ecf6e28312") }
MongoDB server version: 4.4.2
---
The server generated these startup warnings when booting:
        2021-04-04T05:33:44.344-04:00: Access control is not enabled for the database. Read and writ
 configuration is unrestricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to y
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
>
```

MongoDB: Commands

- `show dbs` : display all databases
- `db`: display current database
- `use testDb :`
  - changes current db to testdb (testDb is the name of database)
  - You do not need to create the database before you switch.
  - MongoDB creates the database when you first store data in that database (such as create the first collection in the database).
- `db.myNewCollection2.insertOne( { x: 1 } )`
- `db.myNewCollection3.createIndex( { y: 1 } )`
- `db.createCollection('grade')`
  - Both the insertOne() and the createIndex() operations create their respective collection if they do not already exist
  - MongoDB provides the db.createCollection() method to explicitly create a collection with various options,

MongoDB: Commands

- `Db.getCollectionInfos()`
  - To retrieve the UUID for a collection
  - Collections are assigned an immutable UUID . The collection UUID remains the same across all members of a replica set and shards in a sharded cluster.

MongoDB: CRUD Operations

- CRUD operations *create*, *read*, *update*, and *delete documents*
- Create or insert operations add new documents to a collection.

- MongoDB provides the following methods to insert documents into a collection:
- db.collection.insertOne()
- db.collection.insertMany()

MongoDB: Insert Operation

```
db.inventory.insertOne(
   { item: "canvas",
     qty: 100,
     tags: ["cotton"],
     size: { h: 28, w: 35.5, uom: "cm" } }
)

db.inventory.insertMany(
[
   { item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } },
   { item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } },
   { item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85} }
]
)
```

MongoDB: Query Operation

- Assume the following 'Inventory' collection

```
[
   { item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" },
   { item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "A" },
   { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },
   { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },
   { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" },
   { item: "journal", qty: 25, tags: ["blank", "red"], dim_cm: [ 14, 21 ] },
   { item: "notebook", qty: 50, tags: ["red", "blank"], dim_cm: [ 14, 21 ] },
   { item: "paper", qty: 100, tags: ["red", "blank", "plain"], dim_cm: [ 14, 21 ] },
   { item: "planner", qty: 75, tags: ["blank", "red"], dim_cm: [ 22.85, 30 ] },
   { item: "postcard", qty: 45, tags: ["blue"], dim_cm: [ 10, 15.25 ] }
]
```

MongoDB: Query Operation

- Use MongoDb Compass “filter” option or ‘mongo shell’ to query the collection
  - To display all documents use `{}` in filter option or `db.Inventory.find()` in mongo shell
  - To display all documents where status is ‘A ‘ use : `{status : ‘A'}` OR `db.Inventory.find({status : ‘A'})`
  - To use ‘in’ operator use : `{ status: { $in: [ "A", "D" ] } }`
    `db.Inventory.find({status: { $in: [ "A", "D" ] })`
  - To use ‘and’ operator use : `{ status: "A", qty: { $lt: 30 } }`
    OR `{ $and: [ { status: "A" }, { qty: { $lt: 30 } } ] }`
    `db.Inventory.find({$and: [ { status: "A" }, { qty: { $lt: 30 } } ]})`
  - To use ‘or’ operator use : `{ $or: [ { status: "A" }, { qty: { $gt: 30 } } ] }`
    `db.Inventory.find({$or: [ { status: "A" }, { qty: { $lt: 30 } } ]})`

MongoDB: Query Operation

- To use both ‘or’ and ‘and’ operator use :

```
{item:’/^pa’,{ $or: [ { status: "A" }, { qty: { $gt: 30 } } ] }}
```

- Querying embedded document : `{ size: { h: 14, w: 21, uom: "cm" } }`
- Querying nested field : `{ “size.uom”:”cm”}`
- Querying an array : `{ tags: ["red", "blank"] }`
- Querying an array : `{ tags: { $all: ["red", "blank"] } }`
- To select few fields use :

```
db.inventory.find( { status: "A" }, { item: 1, status: 1 } )
db.inventory.find( { status: "A" }, { status: 0, instock: 0 } )
```

MongoDB: Update Operation

```
db.students.insertMany([
   { _id: 1, test1: 95, test2: 92, test3: 90, modified: new Date("01/05/2020") },
   { _id: 2, test1: 98, test2: 100, test3: 102, modified: new Date("01/05/2020") },
   { _id: 3, test1: 95, test2: 110, modified: new Date("01/04/2020") },
```

- **`db.students.updateOne( { _id: 3 }, [ { $set: { "test3": 98, modified: "$$NOW"} } ] )`**

MongoDB: Update Operation

```
db.students3.insert([
   { "_id" : 1, "tests" : [ 95, 92, 90 ], "modified" : ISODate("2019-01-01T00:00:00Z") },
   { "_id" : 2, "tests" : [ 94, 88, 90 ], "modified" : ISODate("2019-01-01T00:00:00Z") },
   { "_id" : 3, "tests" : [ 70, 75, 82 ], "modified" : ISODate("2019-01-01T00:00:00Z") }
]);
```

```
db.students3.updateMany(
   { },
   [
     { $set: { average : { $trunc: [ { $avg: "$tests" }, 0 ] }, modified: "$$NOW" } },
     { $set: { grade: { $switch: {
                           branches: [
                               { case: { $gte: [ "$average", 90 ] }, then: "A" },
                               { case: { $gte: [ "$average", 80 ] }, then: "B" },
                               { case: { $gte: [ "$average", 70 ] }, then: "C" },
                               { case: { $gte: [ "$average", 60 ] }, then: "D" }
                           ],
                           default: "F"
     } } } }
   ]
)
```

MongoDB: Delete Operation

- `db.Inventory.deleteOne( { "_id" : ObjectId("563237a41a4d68582c2509da") } );`

- `db.Inventory.deleteOne( { "qty:" : { $gt:100} })`

- `db.Inventory.deleteMany( { “qty" : { $gt:100} } )`

- `db.Inventory.remove( { qty: { $gt: 20 } } )`

MongoDB: Aggregation Operation

-
  ```
  db.orders.aggregate([
      { $match: { status: "A" } },
      { $group: { _id: "$cust_id", total: { $sum: "$amount" } } }
  ])
  ```
  - **First Stage**: The `$match` stage filters the documents by the status field and passes to the next stage those documents that have status equal to `"A"`.
  - **Second Stage:** The `$group` stage groups the documents by the `cust_id` field to calculate the sum of the amount for each unique `cust_id`.
- `db.orders.distinct(“cust_id”)`

Views

- A MongoDB view is a queryable object whose contents are defined by an aggregation pipeline on other collections or views.
- MongoDB does not persist the view contents to disk.
- A view's content is computed on-demand when a client queries the view.
- MongoDB can require clients to have permission to query the view.
- MongoDB does not support write operations against views.

View

`Using db.CreateCollection()`

```
db.createCollection(
  "<viewName>",
  {
    "viewOn" : "<source>",
    "pipeline" : [<pipeline>],
    "collation" : {
<collation> }
  }
)
```

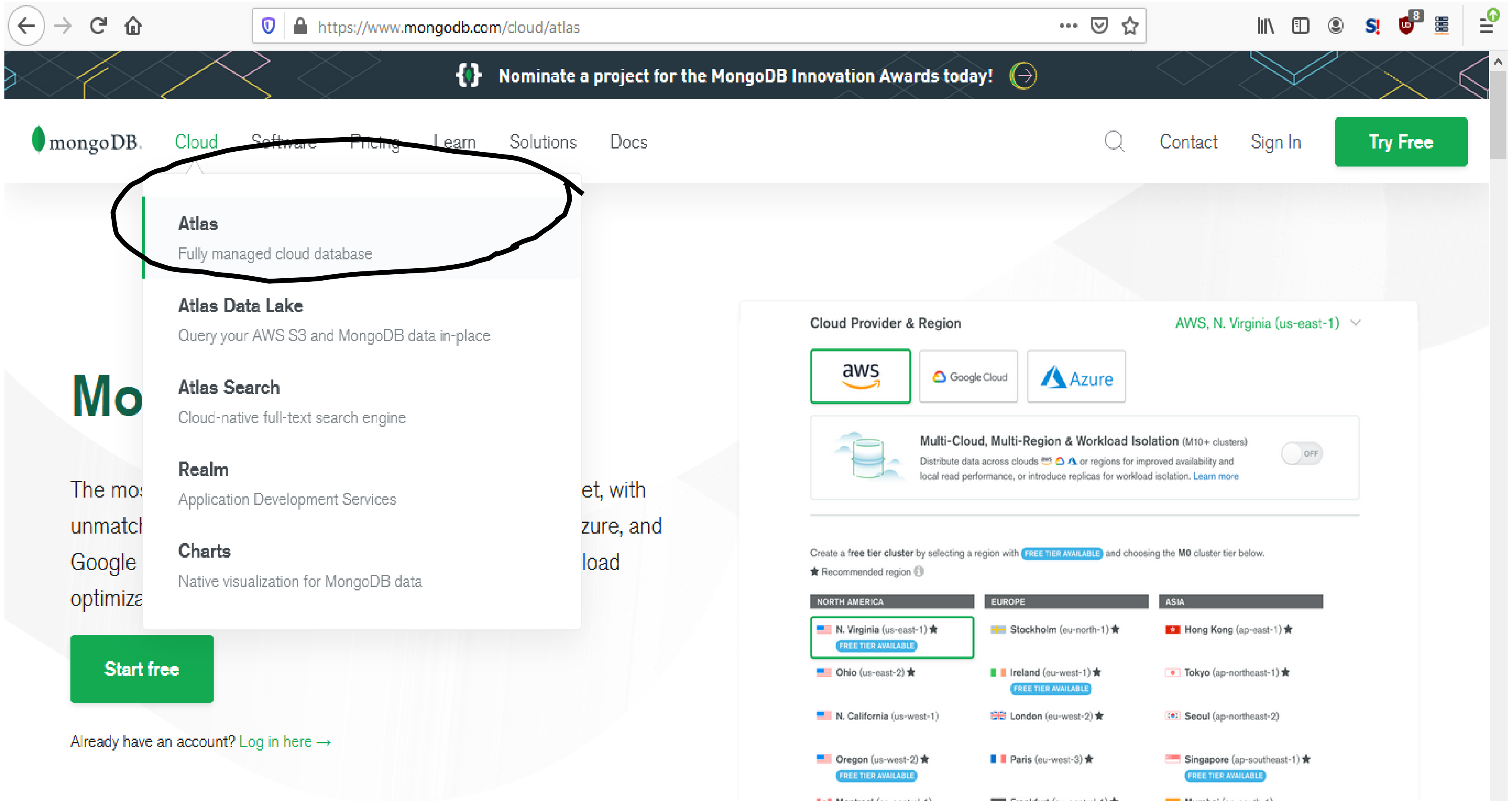
`Using db.CreateView()`

```
db.createView(
  "<viewName>",
  "<source>",
  [<pipeline>],
  {
    "collation" : {
<collation> }
  }
)
```

mongoDB

MONGO ATLAS

Mongo Atlas



https://www.mongodb.com/cloud/atlas

Nominate a project for the MongoDB Innovation Awards today!

mongoDB. Cloud Software Pricing Learn Solutions Docs Contact Sign In **Try Free**

Atlas
Fully managed cloud database

Atlas Data Lake
Query your AWS S3 and MongoDB data in-place

Atlas Search
Cloud-native full-text search engine

Realm
Application Development Services

Charts
Native visualization for MongoDB data

Mo

The mos … et, with
unmatch … zure, and
Google … load
optimiza

Start free

Already have an account? Log in here →

Cloud Provider & Region — AWS, N. Virginia (us-east-1)

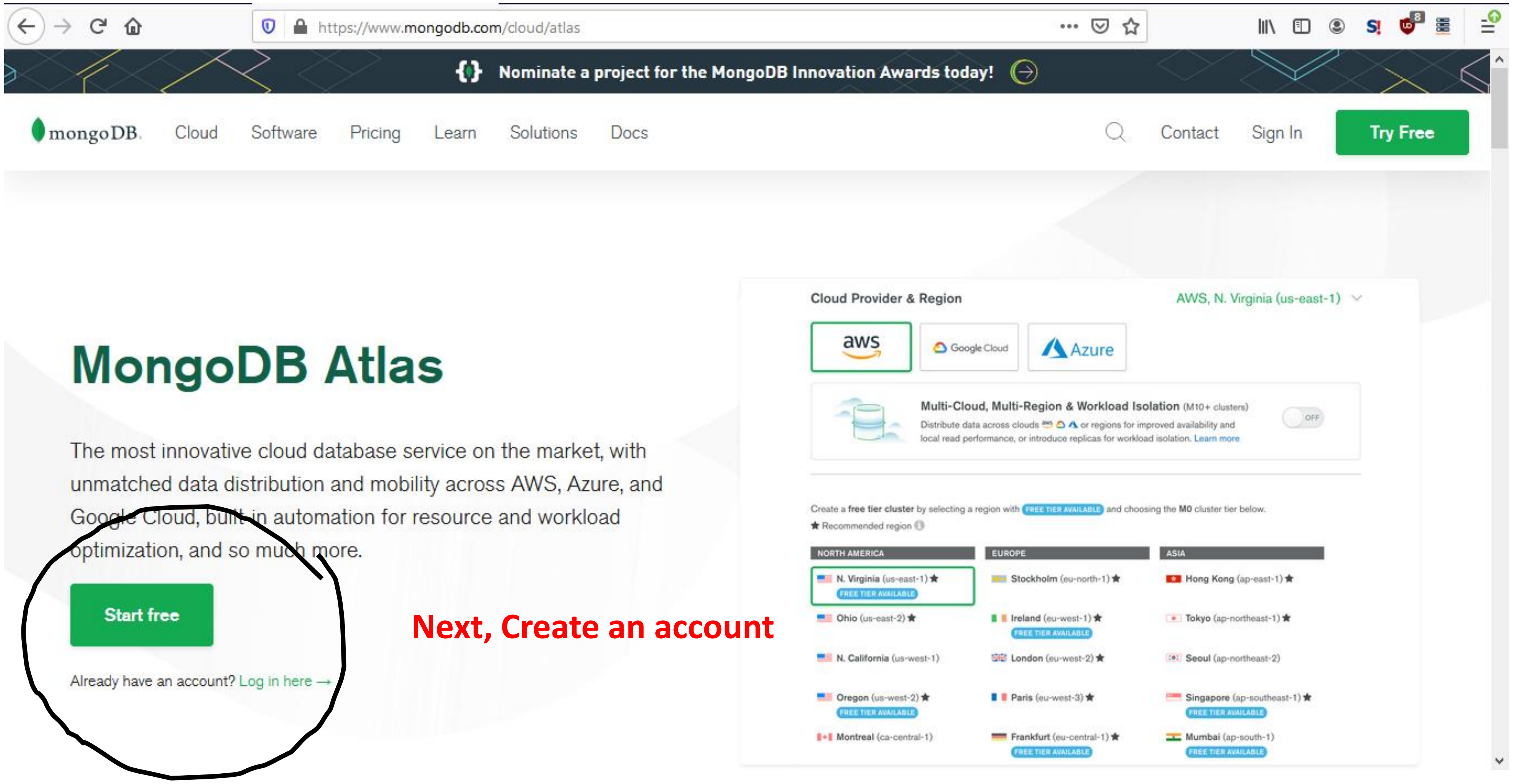
aws | Google Cloud | Azure

Multi-Cloud, Multi-Region & Workload Isolation (M10+ clusters) — OFF
Distribute data across clouds or regions for improved availability and local read performance, or introduce replicas for workload isolation. Learn more

Create a **free tier cluster** by selecting a region with FREE TIER AVAILABLE and choosing the **M0** cluster tier below.
★ Recommended region

NORTH AMERICA	EUROPE	ASIA
N. Virginia (us-east-1) ★ FREE TIER AVAILABLE	Stockholm (eu-north-1) ★	Hong Kong (ap-east-1) ★
Ohio (us-east-2) ★	Ireland (eu-west-1) ★ FREE TIER AVAILABLE	Tokyo (ap-northeast-1) ★
N. California (us-west-1)	London (eu-west-2) ★	Seoul (ap-northeast-2)
Oregon (us-west-2) ★ FREE TIER AVAILABLE	Paris (eu-west-3) ★	Singapore (ap-southeast-1) ★ FREE TIER AVAILABLE

Mongo Atlas



https://www.mongodb.com/cloud/atlas

Nominate a project for the MongoDB Innovation Awards today!

MongoDB Atlas

The most innovative cloud database service on the market, with unmatched data distribution and mobility across AWS, Azure, and Google Cloud, built-in automation for resource and workload optimization, and so much more.

Already have an account? Log in here →

Cloud Provider & Region — AWS, N. Virginia (us-east-1)

Multi-Cloud, Multi-Region & Workload Isolation (M10+ clusters)

Distribute data across clouds or regions for improved availability and local read performance, or introduce replicas for workload isolation. Learn more

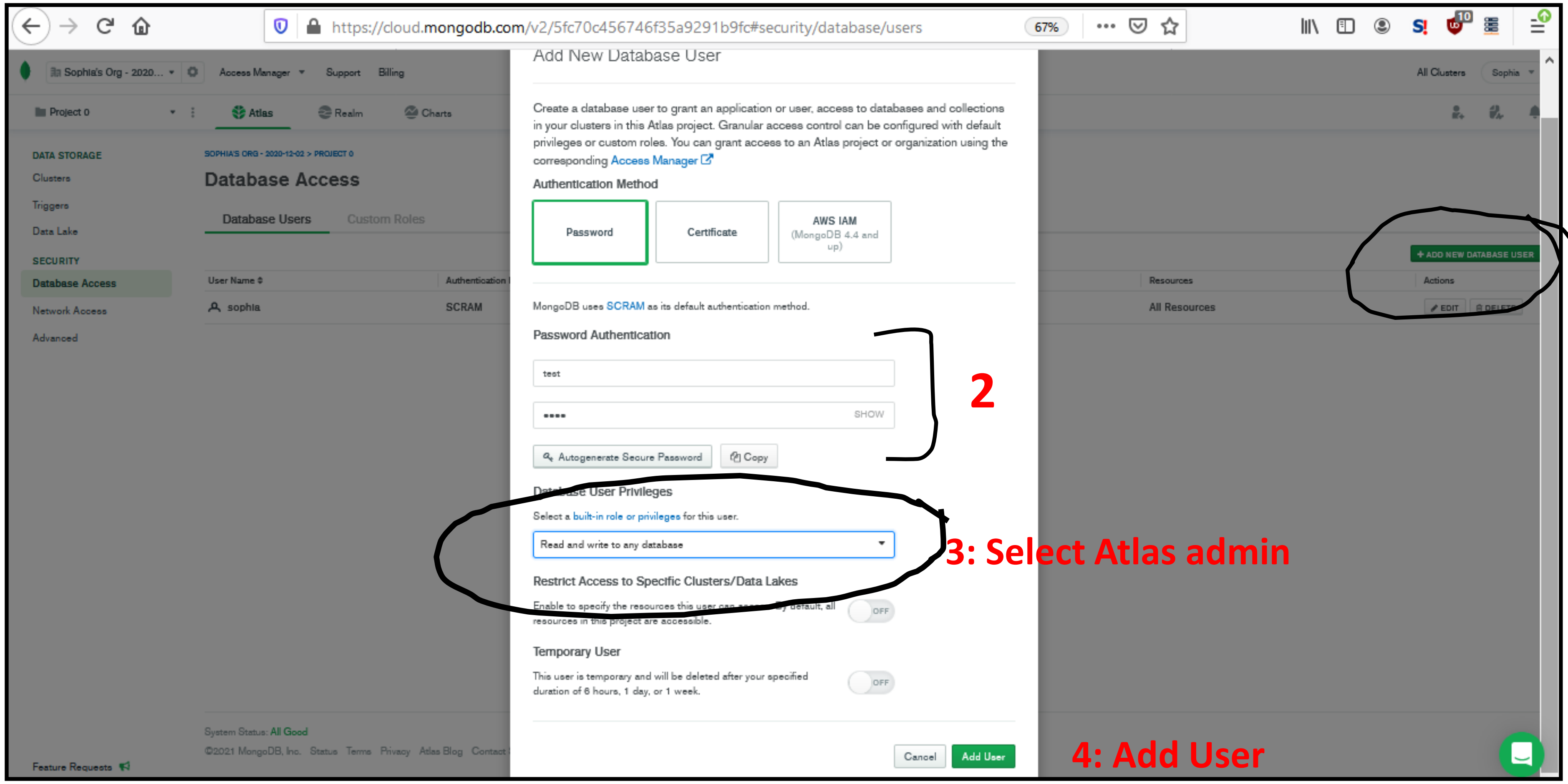
Create a **free tier cluster** by selecting a region with FREE TIER AVAILABLE and choosing the **M0** cluster tier below.

★ Recommended region

NORTH AMERICA	EUROPE	ASIA
N. Virginia (us-east-1) ★ FREE TIER AVAILABLE	Stockholm (eu-north-1) ★	Hong Kong (ap-east-1) ★
Ohio (us-east-2) ★	Ireland (eu-west-1) ★ FREE TIER AVAILABLE	Tokyo (ap-northeast-1) ★
N. California (us-west-1)	London (eu-west-2) ★	Seoul (ap-northeast-2)
Oregon (us-west-2) ★ FREE TIER AVAILABLE	Paris (eu-west-3) ★	Singapore (ap-southeast-1) ★ FREE TIER AVAILABLE
Montreal (ca-central-1)	Frankfurt (eu-central-1) ★ FREE TIER AVAILABLE	Mumbai (ap-south-1) FREE TIER AVAILABLE

Next, Create an account

Mongo Atlas: New User

- Click on Database Access to add new database user.
- 

Add New Database User

Create a database user to grant an application or user, access to databases and collections in your clusters in this Atlas project. Granular access control can be configured with default privileges or custom roles. You can grant access to an Atlas project or organization using the corresponding Access Manager

Authentication Method

Password | Certificate | AWS IAM (MongoDB 4.4 and up)

MongoDB uses SCRAM as its default authentication method.

Password Authentication

test

•••• SHOW

Autogenerate Secure Password | Copy

Database User Privileges

Select a built-in role or privileges for this user.

Read and write to any database

Restrict Access to Specific Clusters/Data Lakes

Enable to specify the resources this user can access. By default, all resources in this project are accessible. OFF

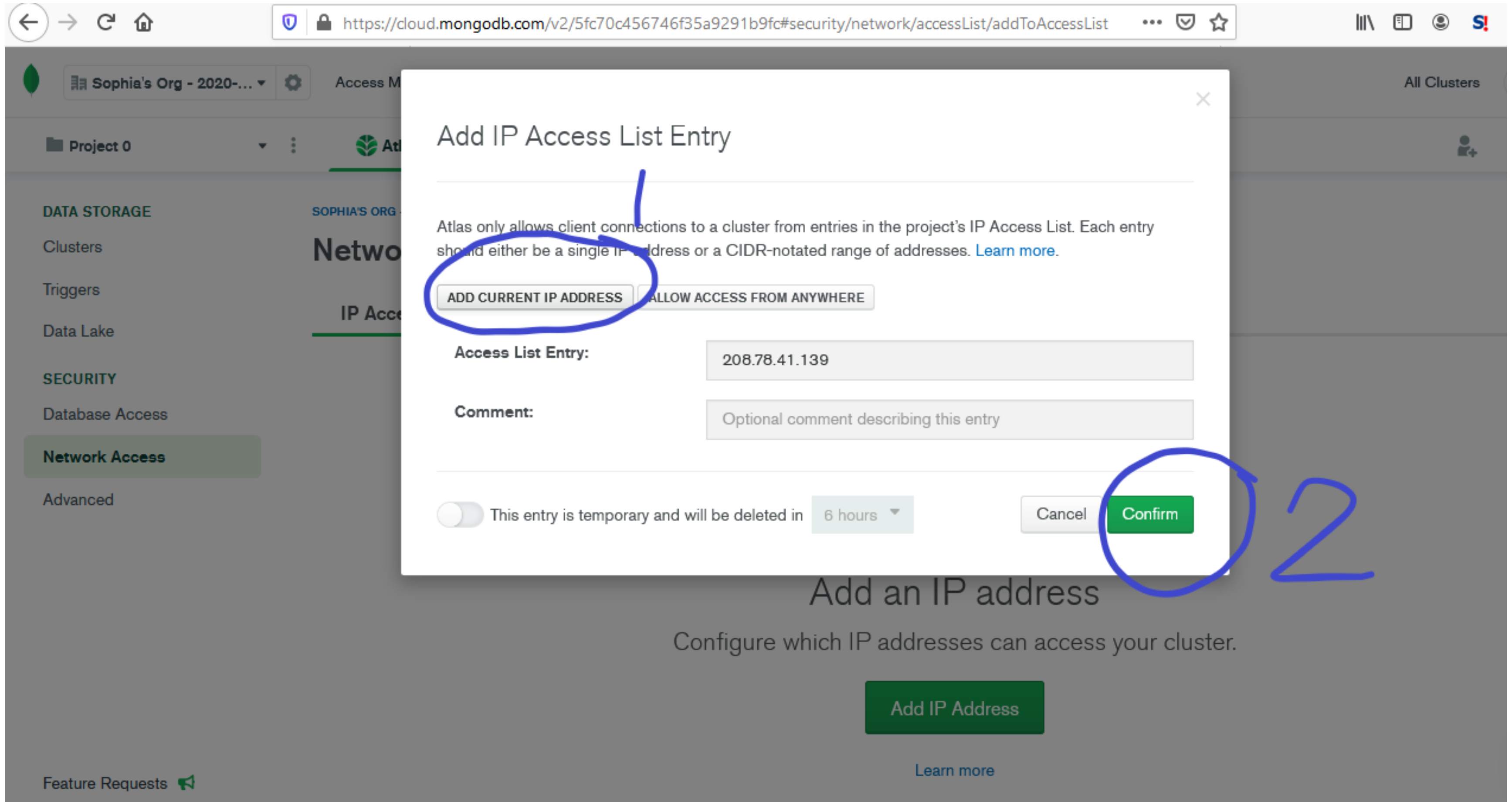
Temporary User

This user is temporary and will be deleted after your specified duration of 6 hours, 1 day, or 1 week. OFF

Cancel | Add User

Mongo Atlas: Network Access

- Click on Network Access to add IP Address.
-



Add IP Access List Entry

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. Learn more.

Access List Entry: 208.78.41.139

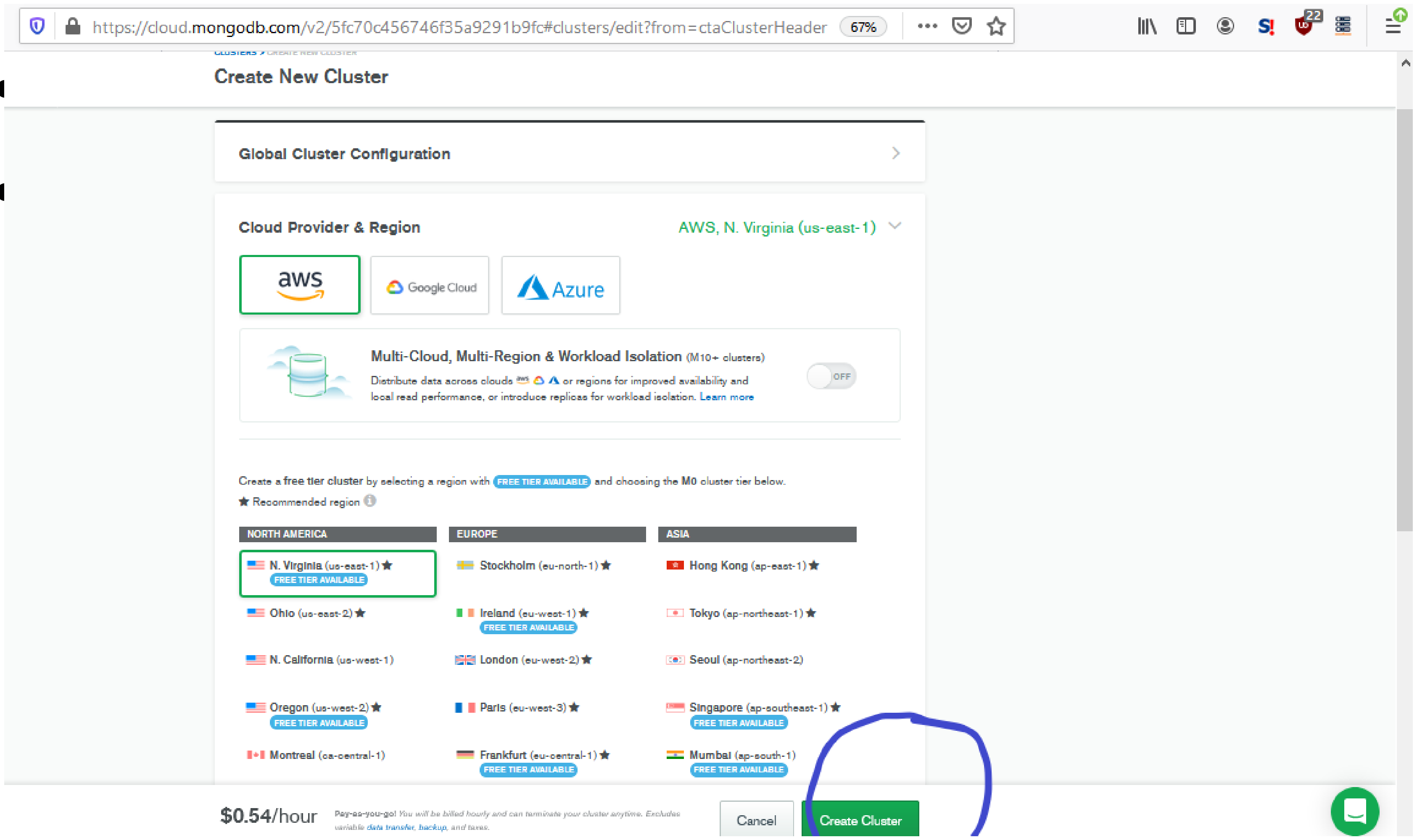
Comment: Optional comment describing this entry

This entry is temporary and will be deleted in 6 hours

Add an IP address

Configure which IP addresses can access your cluster.

Mongo Atlas: Clusters



https://cloud.mongodb.com/v2/5fc70c456746f35a9291b9fc#clusters/edit?from=ctaClusterHeader

CLUSTERS > CREATE NEW CLUSTER

Create New Cluster

Global Cluster Configuration

Cloud Provider & Region — AWS, N. Virginia (us-east-1)

Multi-Cloud, Multi-Region & Workload Isolation (M10+ clusters)

Distribute data across clouds or regions for improved availability and local read performance, or introduce replicas for workload isolation. Learn more

OFF

Create a **free tier cluster** by selecting a region with FREE TIER AVAILABLE and choosing the **M0** cluster tier below.

★ Recommended region

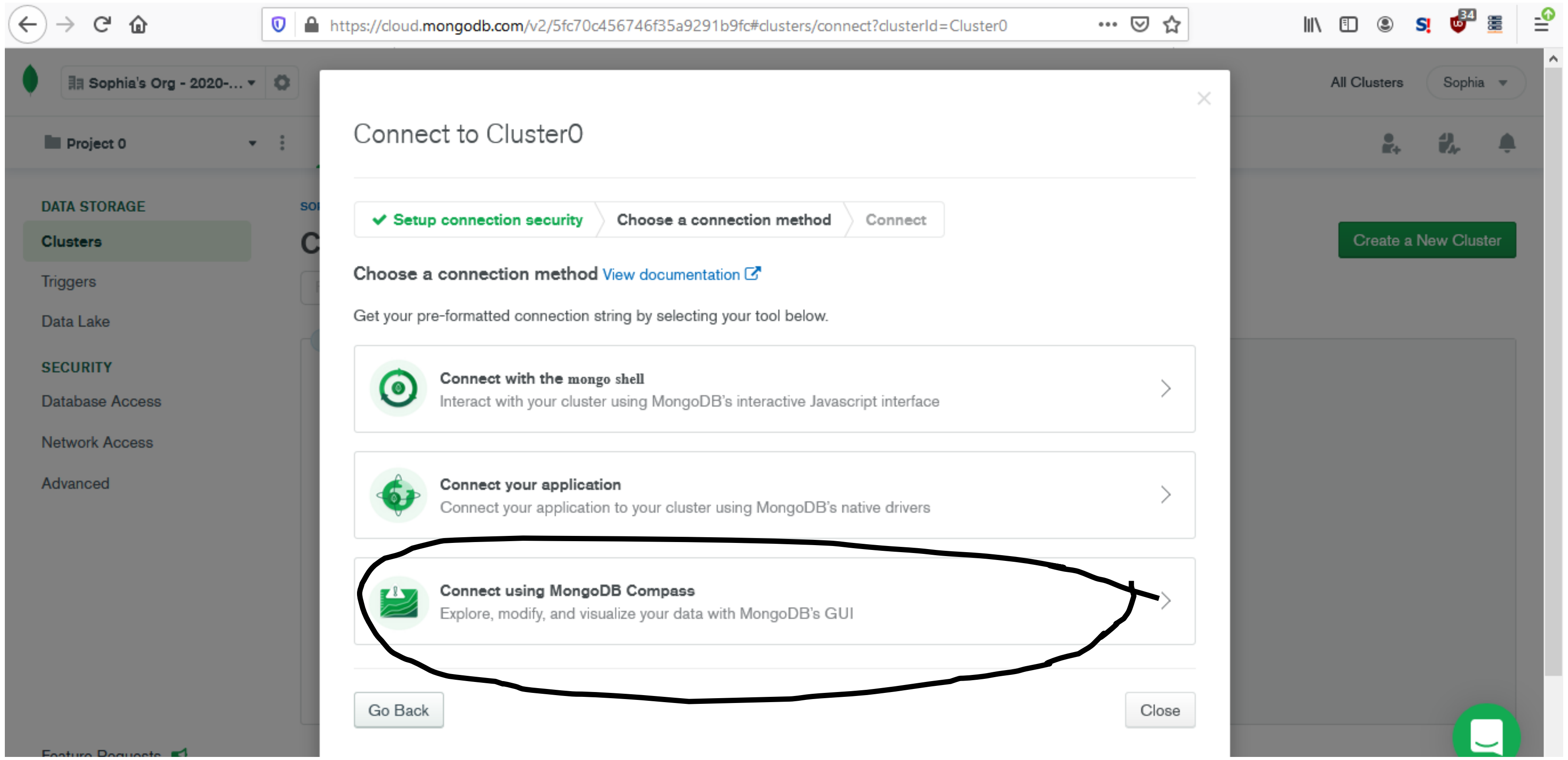
NORTH AMERICA	EUROPE	ASIA
N. Virginia (us-east-1) ★ FREE TIER AVAILABLE	Stockholm (eu-north-1) ★	Hong Kong (ap-east-1) ★
Ohio (us-east-2) ★	Ireland (eu-west-1) ★ FREE TIER AVAILABLE	Tokyo (ap-northeast-1) ★
N. California (us-west-1)	London (eu-west-2) ★	Seoul (ap-northeast-2)
Oregon (us-west-2) ★ FREE TIER AVAILABLE	Paris (eu-west-3) ★	Singapore (ap-southeast-1) ★ FREE TIER AVAILABLE
Montreal (ca-central-1)	Frankfurt (eu-central-1) ★ FREE TIER AVAILABLE	Mumbai (ap-south-1) FREE TIER AVAILABLE

$0.54/hour

Pay-as-you-go! You will be billed hourly and can terminate your cluster anytime. Excludes variable data transfer, backup, and taxes.

Mongo Atlas

- Go to Clusters and click connect .
- Choose Connect using MongoDbCompass



Connect to Cluster0

✓ Setup connection security | Choose a connection method | Connect

Choose a connection method View documentation

Get your pre-formatted connection string by selecting your tool below.

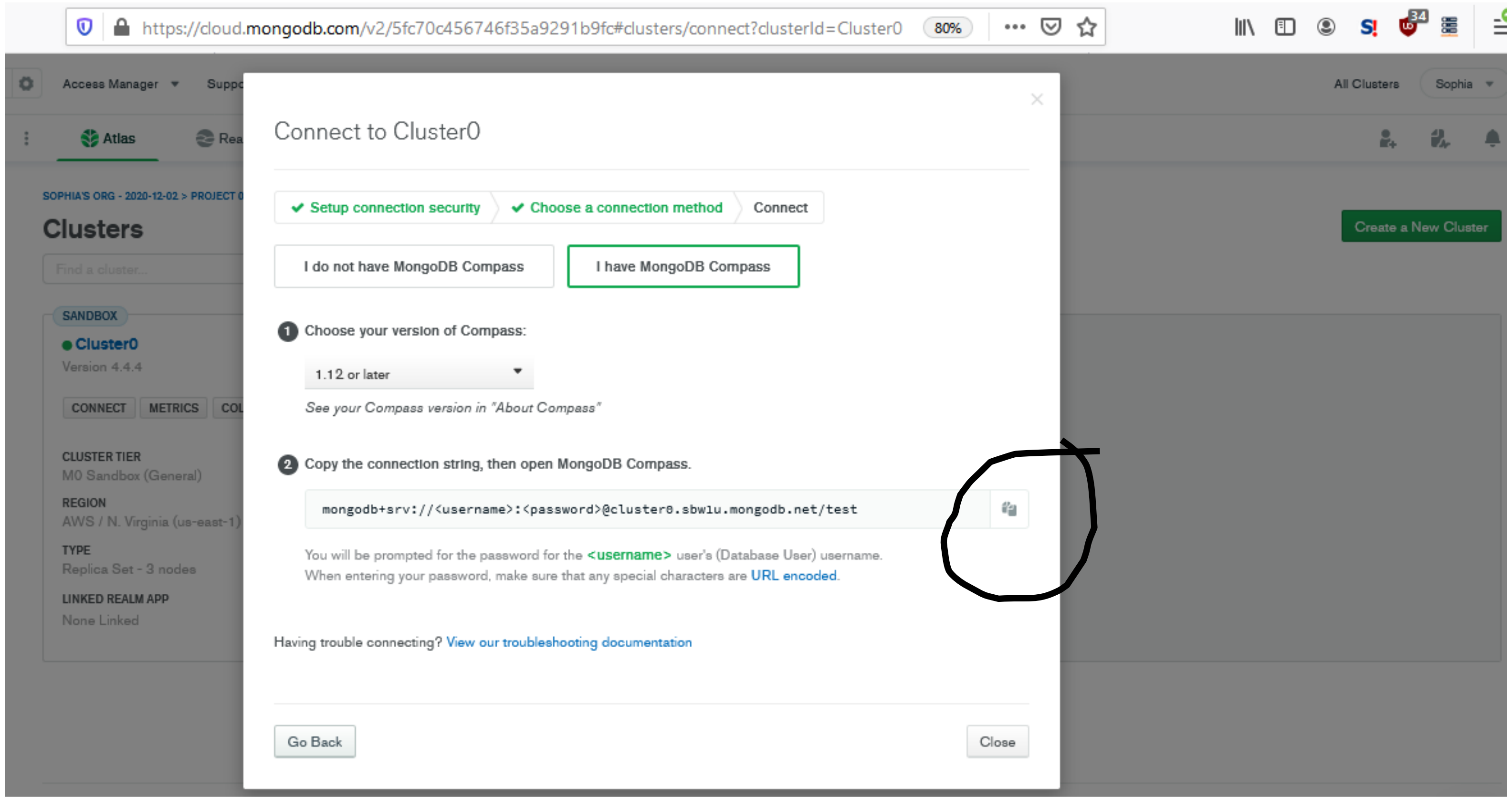
Connect with the mongo shell
Interact with your cluster using MongoDB's interactive Javascript interface

Connect your application
Connect your application to your cluster using MongoDB's native drivers

Connect using MongoDB Compass
Explore, modify, and visualize your data with MongoDB's GUI

Connection String

- Copy the connection string



https://cloud.mongodb.com/v2/5fc70c456746f35a9291b9fc#clusters/connect?clusterId=Cluster0

Connect to Cluster0

✓ Setup connection security ✓ Choose a connection method Connect

I do not have MongoDB Compass | I have MongoDB Compass

1. Choose your version of Compass:

   1.12 or later

   See your Compass version in "About Compass"

2. Copy the connection string, then open MongoDB Compass.

```
mongodb+srv://<username>:<password>@cluster0.sbw1u.mongodb.net/test
```

You will be prompted for the password for the <username> user's (Database User) username. When entering your password, make sure that any special characters are URL encoded.

Having trouble connecting? View our troubleshooting documentation

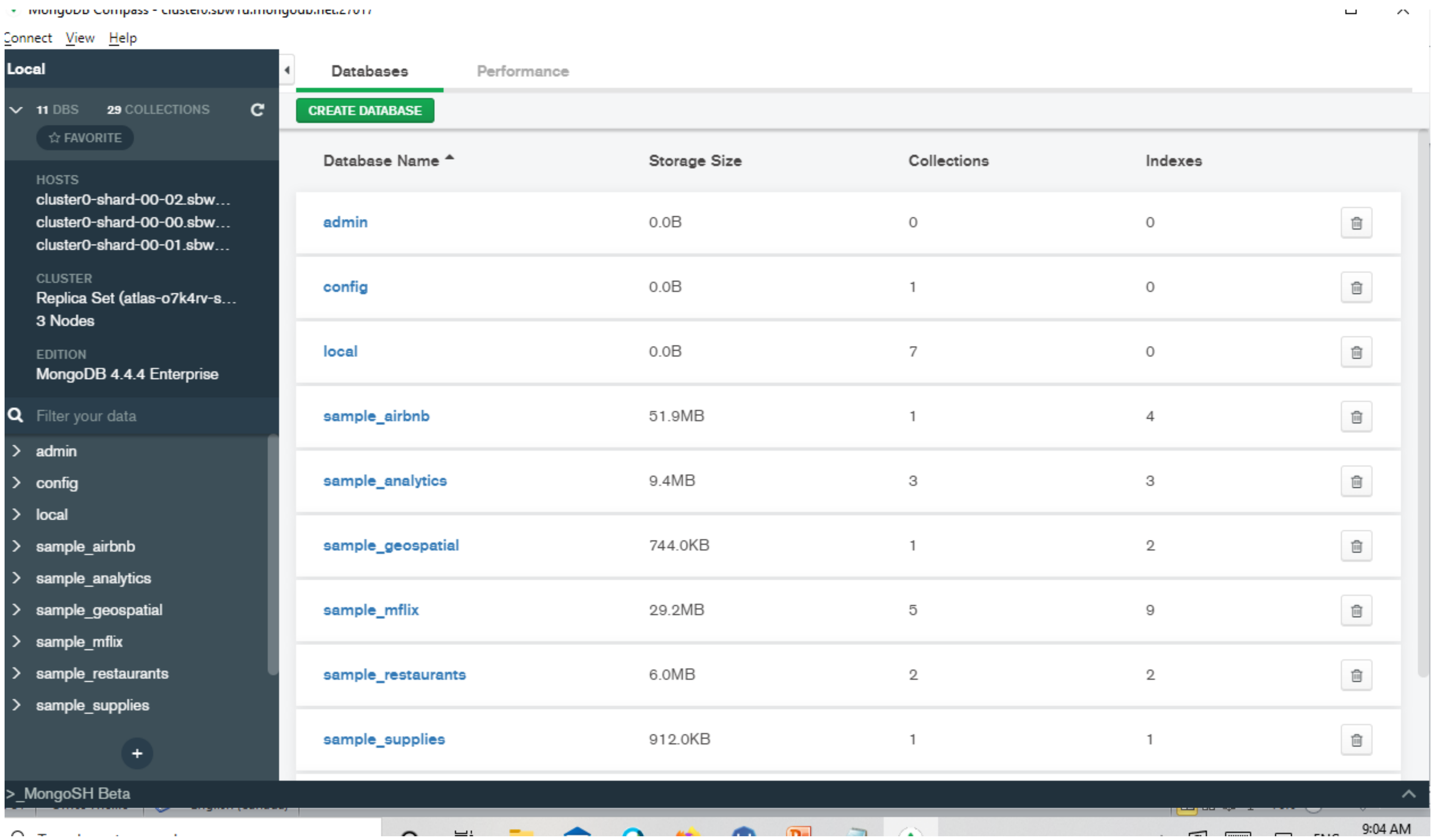
MongoDB Compass

- Copy the connection string and paste it in MongoDb Compass under New Connection.

```
mongodb+srv://<username>:<password>@cluster0.sbw1u.mongodb.net/test
```

- Replace the username and password with actual values(username and password that we created under Database access)

MongoDB Compass



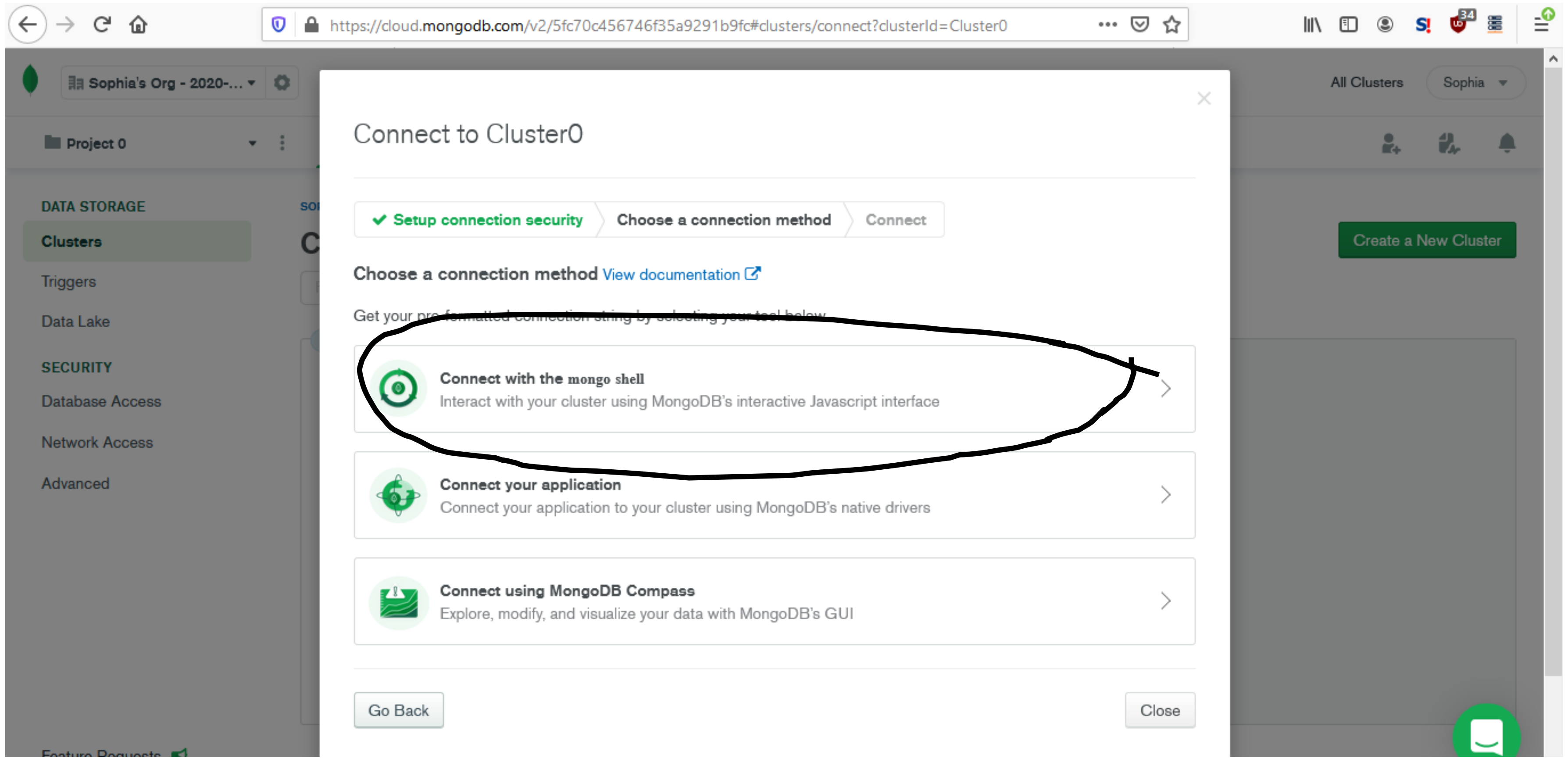
Database Name	Storage Size	Collections	Indexes
admin	0.0B	0	0
config	0.0B	1	0
local	0.0B	7	0
sample_airbnb	51.9MB	1	4
sample_analytics	9.4MB	3	3
sample_geospatial	744.0KB	1	2
sample_mflix	29.2MB	5	9
sample_restaurants	6.0MB	2	2
sample_supplies	912.0KB	1	1

MongoDB Compass

- Use any of the database to run Mongo queries on Mongo Collections.

Mongo Atlas

- Go to Clusters again and click connect .
- This time, choose Connect using Connect with the mongo shell



https://cloud.mongodb.com/v2/5fc70c456746f35a9291b9fc#clusters/connect?clusterId=Cluster0

Connect to Cluster0

✓ Setup connection security | Choose a connection method | Connect

Choose a connection method View documentation

Get your pre-formatted connection string by selecting your tool below.

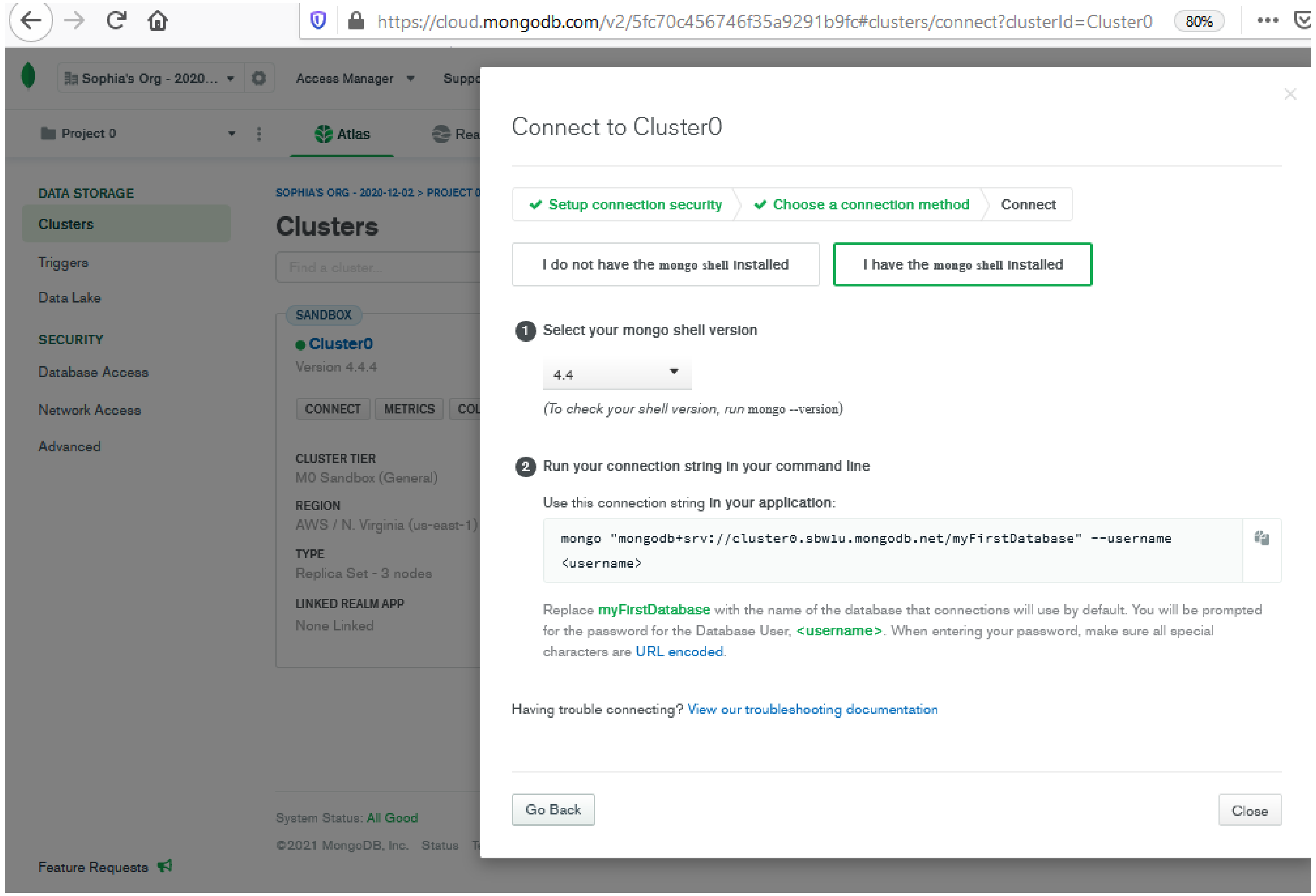
Connect with the mongo shell
Interact with your cluster using MongoDB's interactive Javascript interface

Connect your application
Connect your application to your cluster using MongoDB's native drivers

Connect using MongoDB Compass
Explore, modify, and visualize your data with MongoDB's GUI

Mongo Atlas

- Copy the connection string



https://cloud.mongodb.com/v2/5fc70c456746f35a9291b9fc#clusters/connect?clusterId=Cluster0

Connect to Cluster0

✓ Setup connection security ✓ Choose a connection method Connect

I do not have the mongo shell installed [✓ I have the mongo shell installed]

1 Select your mongo shell version

4.4

(To check your shell version, run mongo --version)

2 Run your connection string in your command line

Use this connection string in your application:

```
mongo "mongodb+srv://cluster0.sbw1u.mongodb.net/myFirstDatabase" --username
<username>
```

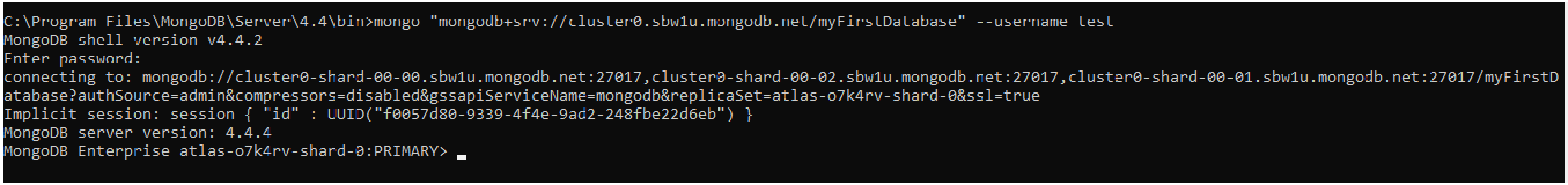
Replace myFirstDatabase with the name of the database that connections will use by default. You will be prompted for the password for the Database User, <username>. When entering your password, make sure all special characters are URL encoded.

Having trouble connecting? View our troubleshooting documentation

Go Back Close

Mongo shell

- Open the command prompt and goto Mongo's bin folder.
- Paste the connection string and replace the username with actual `<username>`
- Run the mongo queries to see all databases that are available and work with any database/collection



```
C:\Program Files\MongoDB\Server\4.4\bin>mongo "mongodb+srv://cluster0.sbw1u.mongodb.net/myFirstDatabase" --username test
MongoDB shell version v4.4.2
Enter password:
connecting to: mongodb://cluster0-shard-00-00.sbw1u.mongodb.net:27017,cluster0-shard-00-02.sbw1u.mongodb.net:27017,cluster0-shard-00-01.sbw1u.mongodb.net:27017/myFirstD
atabase?authSource=admin&compressors=disabled&gssapiServiceName=mongodb&replicaSet=atlas-o7k4rv-shard-0&ssl=true
Implicit session: session { "id" : UUID("f0057d80-9339-4f4e-9ad2-248fbe22d6eb") }
MongoDB server version: 4.4.4
MongoDB Enterprise atlas-o7k4rv-shard-0:PRIMARY> _
```

Future Reading

- In Lynda.com, get the exercise files from
  - Learning MongoDB (Kirsten Hunter)

- MongoDB web site:
  - https://www.mongodb.com
  - MongoDB reference manual: • https://docs.mongodb.com/manual/reference/

- Lynda.com course:
  - Learning MongoDB by Kirsten Hunter

References

- **MongoDB Documentation**
  - **https://docs.mongodb.com/manual/**

Thanks