TASK

Databases and the MongoDB Database

Visit our website

Introduction

WELCOME TO THE DATABASES AND THE MONGODB DATABASE TASK!

Dynamic web applications rely on data. Storing data about your users is a good starting point if you want to make a web application more dynamic. In this task, you will learn about various types of databases. You will also be introduced to MongoDB (a very popular database), and shown how to use MongoDB to create a database-as-a-service (DBaaS) architecture or implementation.

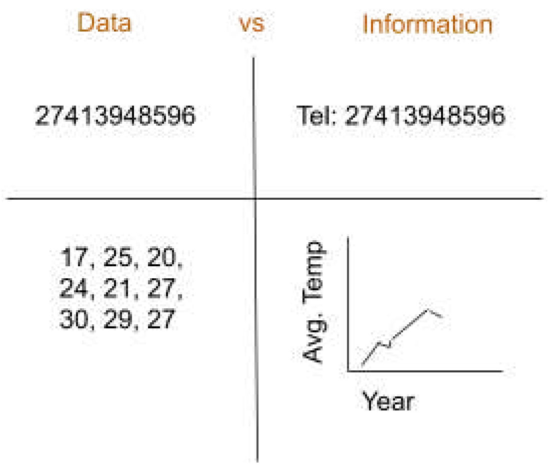
DATABASES

Data is core to software development. Programs are designed to manipulate, create, and visualise data. Therefore, it is important for all software developers and web developers to be able to access, manipulate, and store data.

Databases are used to store data. A database is simply a large container of data with the ability to order the data in multiple ways while providing easy access to the data itself. Web developers often have to manipulate the data stored in databases. For instance, you may need to store your users' usernames, passwords, names, addresses, telephone numbers, etc. Therefore, full-stack web developers need to be able to work proficiently with databases.

DATA VERSUS INFORMATION

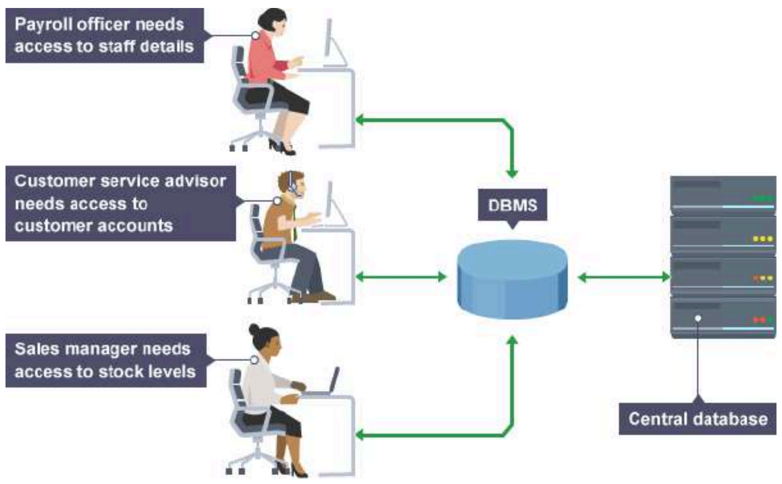
To properly understand databases, you must first understand the difference between data and information. Simply put, data are **raw** facts. The word raw indicates that the facts have not yet been processed to create meaning. Information, on the other hand, is the result of processing raw data to make it meaningful. Data processing can be as simple as organising data to reveal patterns, or as complex as drawing inferences using statistical modelling.



The production of accurate, relevant, and timely information is the key to good decision-making. In turn, good decision-making is the key to a business's survival in a competitive global environment. Timely and useful information requires accurate data, which must be captured properly and stored in a format that is easy to access and process. The data environment should be carefully managed.

DATABASE MANAGEMENT SYSTEM

A database can be thought of as a well-organised electronic filing cabinet where powerful software, known as a database management system (DBMS), helps manage the contents of the cabinet. A database management system is a collection of programs that manage the database structure and control access to the data stored in the database.


Payroll officer needs access to staff details
Customer service advisor needs access to customer accounts
Sales manager needs access to stock levels
DBMS
Central database

Bell, C. (2015). *Database Management System (DBMS) Relationships* [Illustration]. Chrisbell.com. **https://www.chrisbell.com/SNHU/IT-510-Advanced-Information-Technology/database-management-system-DBMS-relationships.php**

The illustration above shows how the DBMS acts as an intermediary between the user and the database, handling requests and translating them into operations. It hides the database's complexity, allowing data to be shared among multiple users and integrating different views into a single data repository.

The DBMS helps make data management much more efficient and effective; and provides advantages such as:

- **Improved data sharing:** the DBMS helps create an environment in which end users have better access to more and better-managed data.
- **Better data integration:** an integrated view of the organisation's operations and a clearer view of the big picture is promoted by wider access to well-managed data.

- **Minimised data inconsistency:** data inconsistency occurs when different versions of the same data appear in different places. A properly designed database greatly reduces the probability of data inconsistency.

- **Improved data access:** a query is a specific request for data manipulation (e.g. to read or update the data) sent to the DBMS. The DBMS makes it possible to produce quick answers to spur-of-the-moment queries.

- **Improved decision-making:** better-quality information (on which to base decisions) is generated, due to better-managed data and improved data access.

- **Increased end-user productivity:** the availability of data and the tools that transform data into usable information encourage end users to make quick, informed decisions.

TYPES OF DATABASES

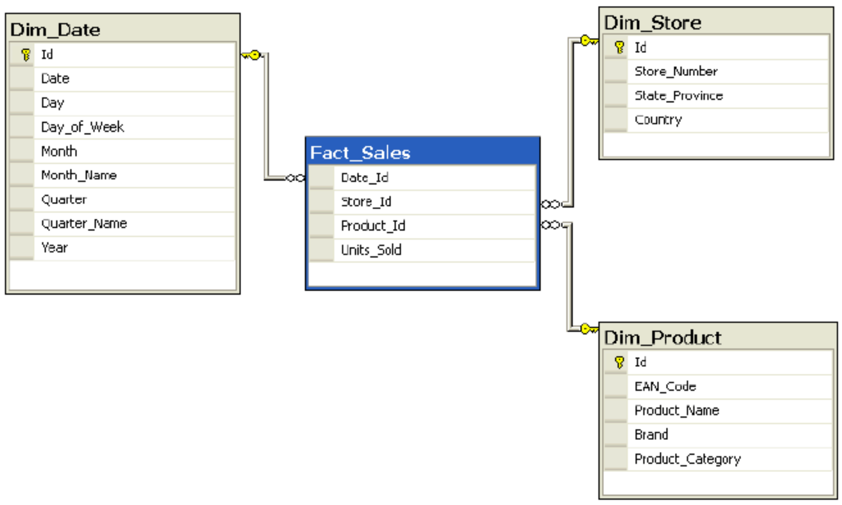
Databases can be classified based on users, location, type, usage, and structure.

A **single-user database** supports one user at a time, like a desktop database. A **multi-user database** allows simultaneous users; a workgroup database supports fewer than 50 users, while an enterprise database supports many users across departments.

By location, databases can be **centralised** (data at one site) or **distributed** (data spread across multiple sites). Based on usage, **operational databases** support day-to-day activities, while **analytical databases** store historical data for decision-making.

Databases also differ in structure. **Unstructured data** are raw, as collected, while **structured data** are organised for specific use, like in a spreadsheet for calculations.

The two most common types are **relational databases**, which store entities and relationships (as shown in the entity-relationship diagram below), and **NoSQL databases**:



N.d. (2023). *Star schema*. Wikipedia, The Free Encyclopedia.
https://en.wikipedia.org/wiki/Star_schema#/media/File:Star-schema-example.png

In a relational database, an entity is a table that stores all the data about a certain thing. For example, you may want to store information about all your customers in a database. You would then create a customer table (customer entity), which could look something like the one shown below:

C_NAME	C_PHONE	C_ADDRESS	C_POST CODE	A_NAME	A_PHONE	TP	AMT	REN
Alfred Smith	082 345 2341	207 Willow St, Port Elizabeth	6390	Leah Hahn	084 259 2073	T1	R100.00	05-Apr-2021
Kathy Dunne	083 567 9012	556 Bad St, Cape Town	7100	Alex Alby	085 785 3938	S2	R250.00	16-Jun-2021
Paul Farris	076 782 1232	2148 High St, Benoni	1522	Leah Hahn	084 259 2073	T2	R850.00	22-Sep-2021

Customer entity fields:

`C_NAME` = customer name

`C_PHONE` = customer phone

`C_ADDRESS` = customer address

`C_POSTCODE` = customer postcode

`A_NAME` = agent name

`A_PHONE` = agent phone

`TP` = insurance type

`AMT` = insurance policy amount in thousands of R

`REN` = insurance renewal date

The `CUSTOMER` table contains three records. Each record is composed of nine fields: `C_NAME`, `C_PHONE`, `C_ADDRESS`, `C_POSTCODE`, `A_NAME`, `A_PHONE`, `TP`, `AMT`, and `REN`. Each record describes a specific customer; each customer is an instance of the customer entity.

In a store database, you'd have a product table and a customer table to store product and customer info. The database would also track relationships, such as which products a customer bought on a specific date. Details on how this is implemented in a relational database are beyond this course's scope.

NoSQL databases

Relational databases struggle with performance as data volume grows, especially for large-scale web applications like Amazon or Google. This led to the creation of NoSQL databases, which you use when searching on Amazon, watching YouTube, or messaging on Facebook. NoSQL databases typically have the following characteristics:

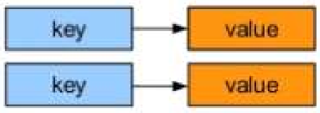
- They are not based on the relational model.
- They support distributed database architectures, i.e. servers in different areas.
- They provide high scalability, high availability, and fault tolerance.
- They support very large amounts of sparse data (data in which there are a

lot of attributes, but many of these are not populated).

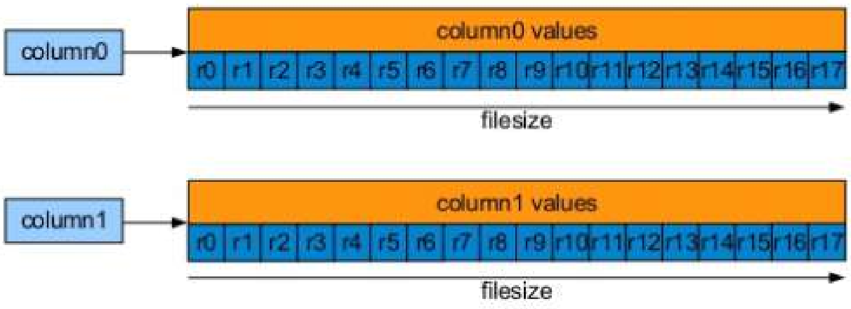
- They are geared toward performance rather than transaction consistency.

There are several types of NoSQL databases. Some of these are briefly described below:

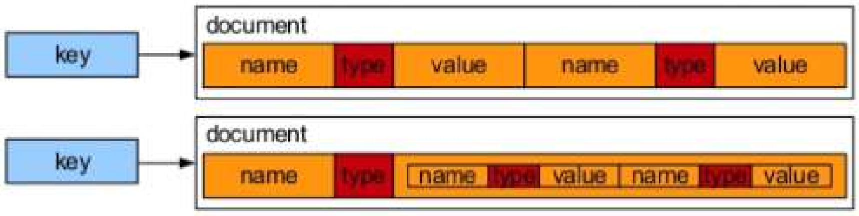
1. **Key-value store databases**: This is the simplest form of the NoSQL database. Every item in the database is stored as a key (used to identify the value) and its value.



2. **Column-oriented databases**: Like key-value store databases, a key is used to identify values, but instead of the key identifying only one value, it can be used to identify multiple values.

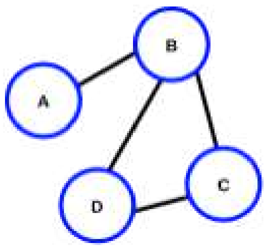

column0
column0 values
r0 r1 r2 r3 r4 r5 r6 r7 r8 r9 r10 r11 r12 r13 r14 r15 r16 r17
filesize
column1
column1 values
r0 r1 r2 r3 r4 r5 r6 r7 r8 r9 r10 r11 r12 r13 r14 r15 r16 r17
filesize

3. **Document-store databases**: With this type of database, a key is used to identify a particular document. Documents are stored in recognised formats like XML, JSON, PDF, etc.



4. **Graph databases**: This database uses the graph data structure to store data. Graphs contain nodes (A, B, C, and D in the image below) and edges (the lines connecting the nodes, so AB, BD, BC, and DC in the image below). In graph databases, nodes are objects and edges are the relationships between these objects. Social networking applications, like Facebook, often store data using graph databases because this model is good for tracking

the relationships between objects.



5. **Object-oriented databases**: These databases combine object-oriented programming and database principles. These databases are tied to specific programming languages.

MONGODB

In this bootcamp, you will be working with MongoDB, a document store and NoSQL database. MongoDB is made up of collections and documents.

- **Collection:** A collection is a group of documents. It is similar to an entity or table when working with relational databases.
- **Documents:** In relational databases, records are stored in tables; for instance, Alfred Smith would be a record in the `CUSTOMER` table. In contrast, MongoDB stores data in documents rather than rows. MongoDB uses BSON (Binary JSON) documents, which include **type** information, making it faster and more efficient than standard JSON files that require parsing (see an example below):

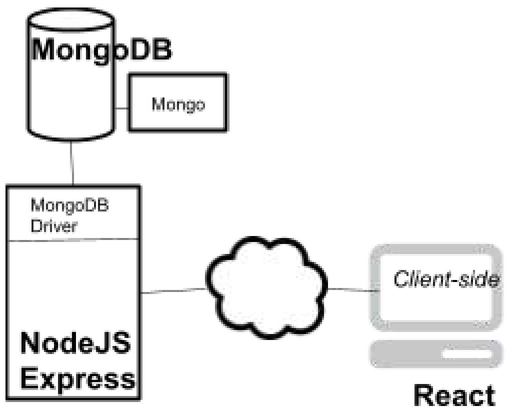
```
{                                                |
  "name": "Joe Drumgoole",                       | Strings
  "title": "Director of Developer Advocacy",     |
                                                 |
  "Address": {                                   | Nested Document
    "address1": "Latin Hall",                    |
    "address2": "Golden Lane",                   |
    "eircode": "D09 N623"                        |
  },                                             |
                                                 |
  "expertise": ["MongoDB", "Python",            | Array
                "Javascript"],                  |
                                                |
  "employee_number": 320,                       | Integer
                                                |
  "location": [53.34, -6.26]                    | Geo-spatial Coordinates
}                                               |
```

Extra resource

The following **playlist** from the MongoDB YouTube channel has a few reasonably short videos that can assist with getting the full picture of what MongoDB is all about and how it compares to relational databases such as SQL-based databases.

MONGODB IN A FULL-STACK WEB APPLICATION

Before we start using MongoDB, it is important to first see how it fits into full-stack web development and what other tools are needed to get it to work properly.



Consider the image above. The app uses Node.js for server-side logic and request routing. For persistent data, it connects to a MongoDB database. MongoDB's admin shell, Mongo, allows command-line database interactions using MongoDB's query language.

For Node.js to be able to communicate with MongoDB, it also makes use of *MongoDB drivers*. The official MongoDB driver can be installed using NPM (more on this later).

MONGODB AS A SERVICE

Previously, running a database required dedicated servers and administrators. Today, the database as a service (DBaaS) offers an alternative. With cloud computing's rise, developers can use cloud-hosted databases like MongoDB Atlas instead of managing their own servers. This approach offers several benefits:

1. It is often cheaper than having your own database server, because you only pay for what you use.
2. The cloud service provider deals with all the hassle of managing the configuration, backup, maintenance, and security of the database server.
3. It is quick and easy to start using a database with minimal configuration.

In this task, you will create your first database using MongoDB. Before you can do this, though, you are first going to:

1. download and install MongoDB on your local machine so that you can use Mongo, the administrative shell;
2. use Atlas to create and host a MongoDB on the cloud; and
3. use Mongo to access and manipulate your database cluster on Atlas.

INSTALL MONGODB

For this course, we will be installing MongoDB's Community Server. This edition provides everything you need to learn MongoDB and is a free alternative to the Enterprise Server. Follow the steps linked below to get started with MongoDB.

- **MongoDB Community Edition Installation Steps**

The steps to set up and configure MongoDB differ slightly depending on the operating system (OS) you are using. The installation instructions given above contain instructions for running MongoDB on your specific OS.

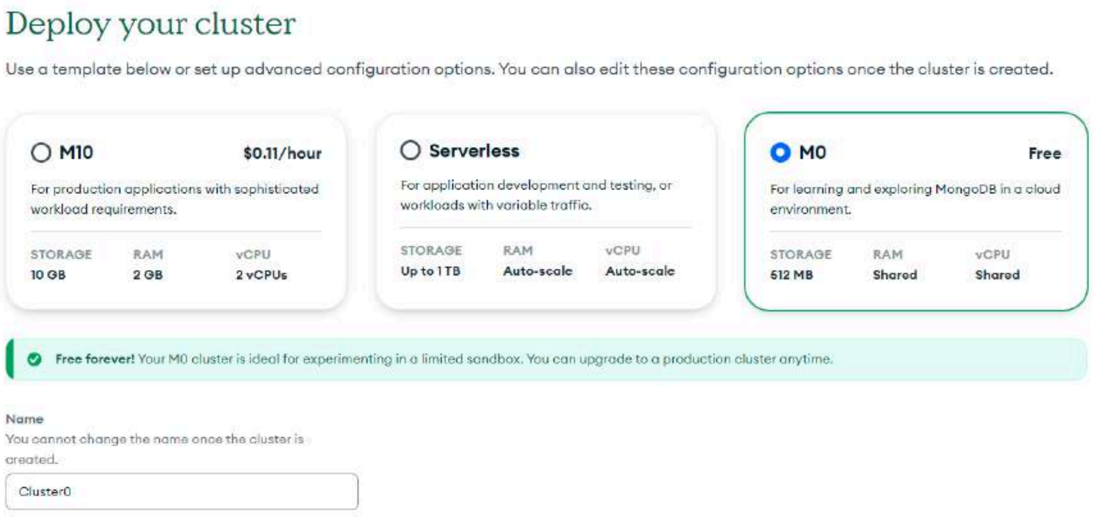
Your Mongo shell is now ready to be used to connect to a database server. MongoDB Atlas will provide the platform and infrastructure we need for a database server on the cloud.

SETUP MONGODB ATLAS

As previously stated, in this bootcamp, we will be using MongoDB's DBaaS solution: Atlas. To get a quick (two-minute) overview of what Atlas is and why we are using it, watch this **short video**.

To configure MongoDB Atlas, do the following:

- Go **here** and enter your information to get started with Atlas.
- You will then be taken to the "Database Deployments" page:
    - Click on the "Build a Database" button on screen.
    - You'll be redirected to a new screen.
    - Under "Cloud provider and Region", select AWS and any free tier region.
    - Under "Cluster Tier", select the free M0 option.
    - You can rename your cluster under "Cluster Name".



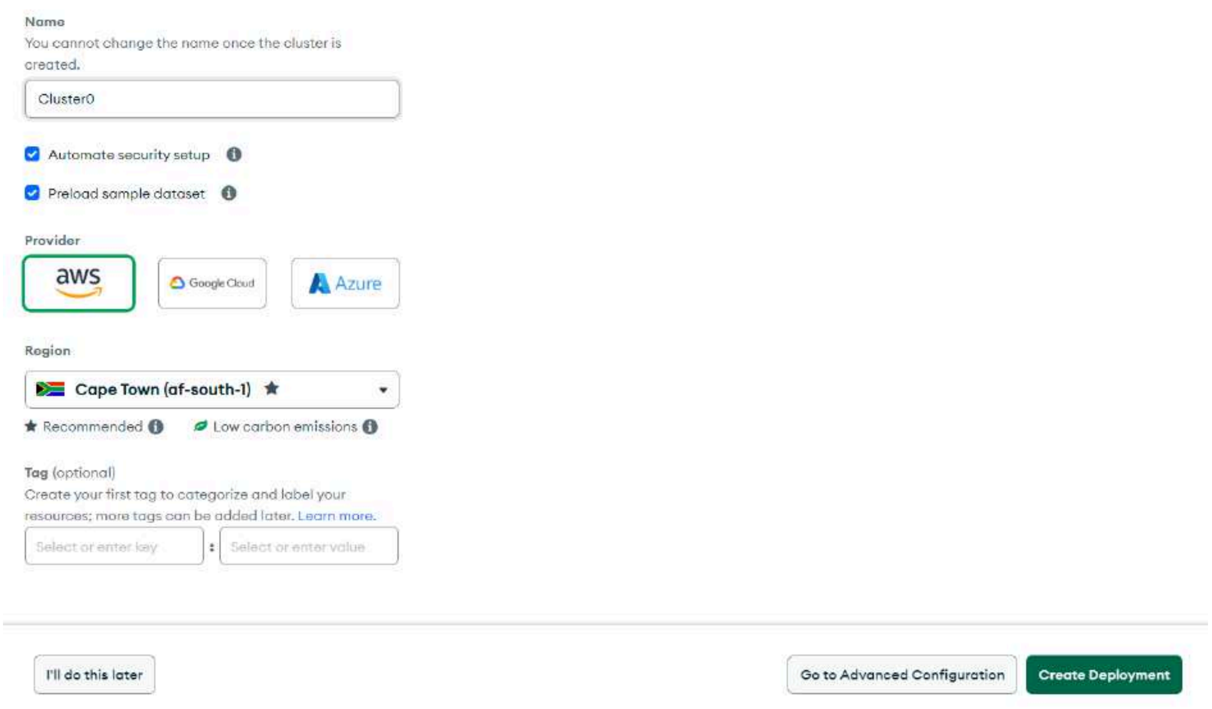
Deploy your cluster

Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.

M10	$0.11/hour	For production applications with sophisticated workload requirements.	STORAGE 10 GB, RAM 2 GB, vCPU 2 vCPUs
Serverless		For application development and testing, or workloads with variable traffic.	STORAGE Up to 1 TB, RAM Auto-scale, vCPU Auto-scale
[✓ M0]	Free	For learning and exploring MongoDB in a cloud environment.	STORAGE 512 MB, RAM Shared, vCPU Shared

Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Name
You cannot change the name once the cluster is created.
Cluster0


Name
You cannot change the name once the cluster is created.
Cluster0

☑ Automate security setup
☑ Preload sample dataset

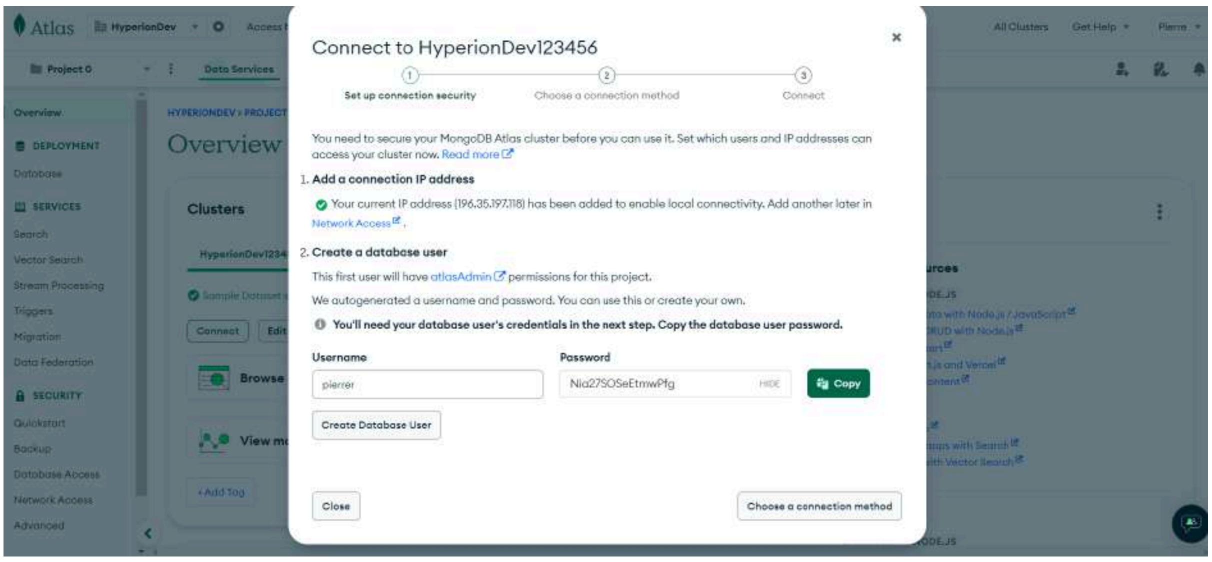
Provider
aws | Google Cloud | Azure

Region
Cape Town (af-south-1) ★
★ Recommended   Low carbon emissions

Tag (optional)
Create your first tag to categorize and label your resources; more tags can be added later. Learn more.
Select or enter key : Select or enter value

Click on the “Create Deployment” button at the bottom of the page to create your cluster.

Once you have created your cluster, you will be taken to a page similar to the one shown below. You can change the username and other settings for your database user as shown in the image.


Connect to HyperionDev123456

1 Set up connection security — 2 Choose a connection method — 3 Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. Read more

1. Add a connection IP address

✓ Your current IP address (196.35.197.118) has been added to enable local connectivity. Add another later in Network Access.

2. Create a database user

This first user will have atlasAdmin permissions for this project.

We autogenerated a username and password. You can use this or create your own.

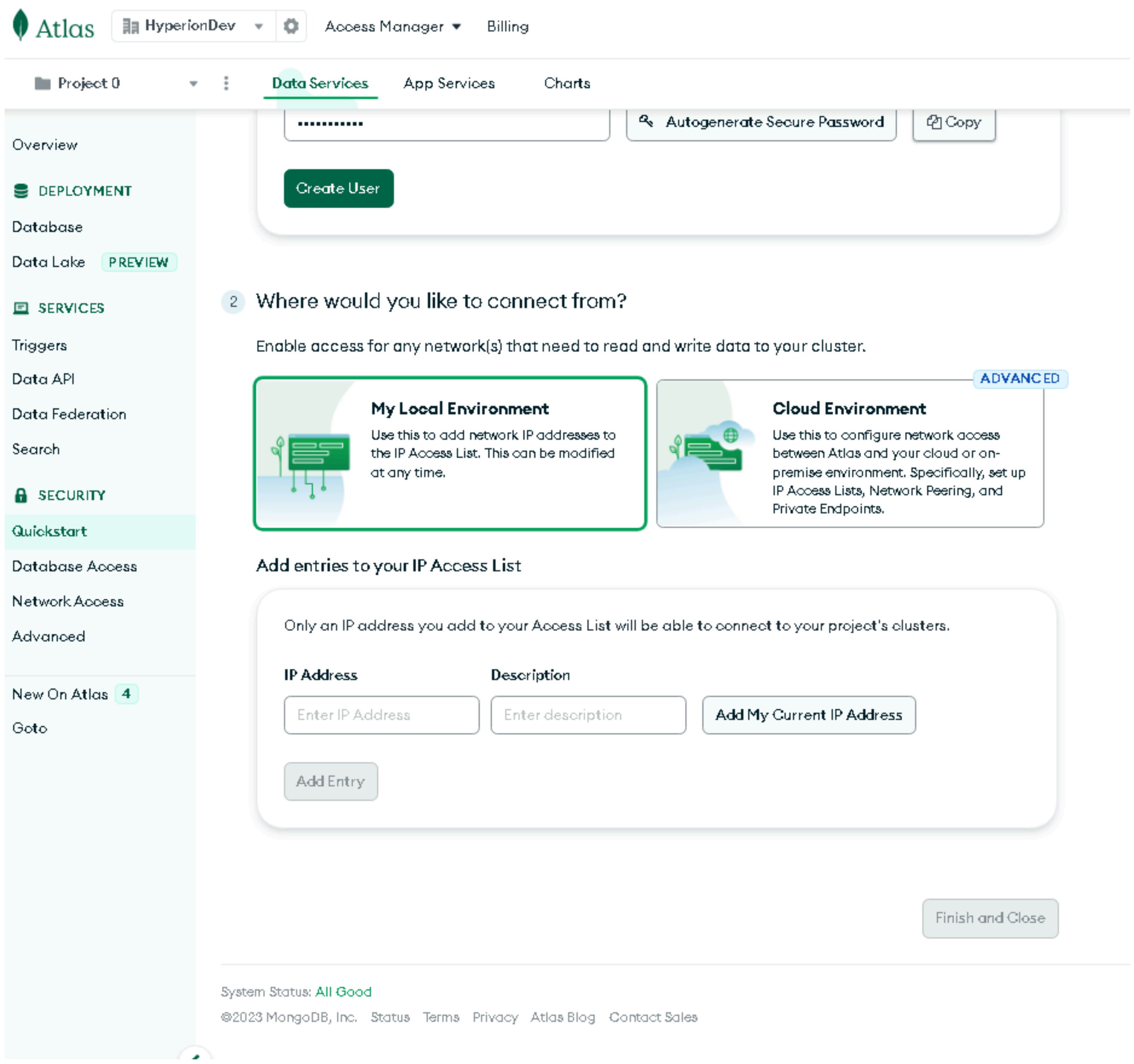
You'll need your database user's credentials in the next step. Copy the database user password.

Username: pierrer
Password: Nia27SOSeEtmwPfg

You have successfully created a database cluster. You will now use the administrative shell, Mongo, to add a database to this cluster. Before you do this, though, there are some security settings that you need to tweak to make sure that you can access the Atlas cluster from your computer.

Security settings

One of the ways that MongoDB ensures security is by only allowing certain machines (IP addresses) access to your cluster. You can whitelist your IP by scrolling down on the “Quickstar”’ tab and clicking “Add my current IP Address”. After also adding a user, click “Finish and Close” at the bottom to continue.



2 Where would you like to connect from?

Enable access for any network(s) that need to read and write data to your cluster.

My Local Environment
Use this to add network IP addresses to the IP Access List. This can be modified at any time.

Cloud Environment
Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.

Add entries to your IP Access List

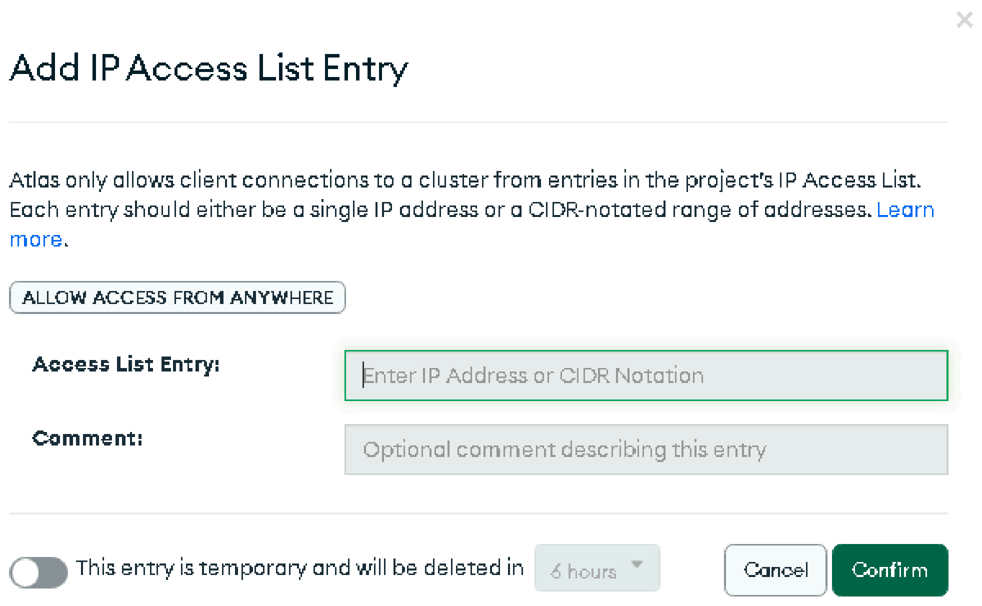
Only an IP address you add to your Access List will be able to connect to your project's clusters.

System Status: All Good

©2023 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

The IPs that are allowed to access your cluster are listed under the “Network Access” tab in the left-hand navigation bar. If you currently have a dynamic IP address, you’ll need to allow access from anywhere.

To do so, click on the “Network Access” tab and then click on the “ADD IP ADDRESS” button. The following pop-up will then appear.

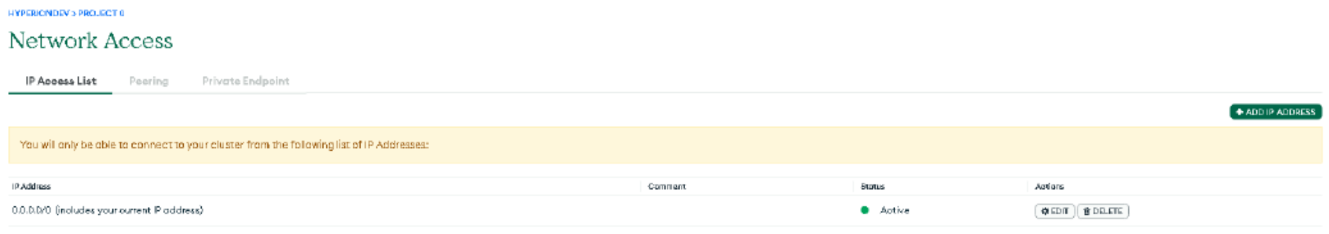

Add IP Access List Entry

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.

ALLOW ACCESS FROM ANYWHERE
Access List Entry:
Enter IP Address or CIDR Notation
Comment:
Optional comment describing this entry
This entry is temporary and will be deleted in
6 hours
Cancel
Confirm

Click “ALLOW ACCESS FROM ANYWHERE” and click “Confirm”.

The following entry should appear in your “Network Access” tab, confirming you’ve allowed access from anywhere:


HYPERIONDEV > PROJECT 0

Network Access

You will only be able to connect to your cluster from the following list of IP Addresses:

IP Address	Comment	Status	Actions
0.0.0.0/0 (includes your current IP address)		● Active	EDIT DELETE

It is not good practice to allow all IP addresses to access your database for obvious security reasons, but for the purposes of the next few tasks, we are going to ask you to do this. The reason for this is that you will need to give a mentor access to your database, and you won’t know their IP address ahead of time. In professional practice, however, it is advisable to have a limited IP whitelist.

Take note:

Configure the IP whitelist when deploying your app. Your Express app will access the database, and if deployed to the cloud (like AWS), the web server's IP address may change. If it isn't added to the Atlas IP whitelist, your app won't communicate with the database.

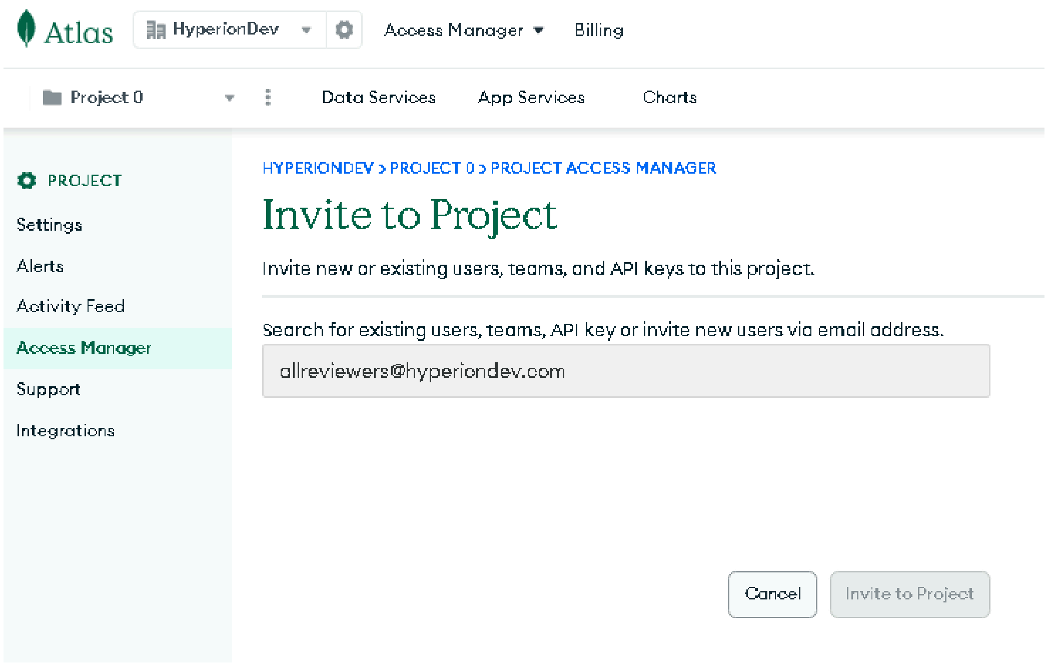
To address this problem, you can use Render’s Static Outbound IP Addresses. According to an **article by Render**, “You can use these addresses to connect to IP-restricted environments outside your Render network.” Outbound requests from those apps originate from a set of static IP addresses, which allows you to securely communicate with IP whitelisted services on-premise (hosted on-site) or on other networks.

If your machine is protected by a firewall, you also have to ensure that this doesn't block access to Atlas. Atlas servers run on ***port 27017*** on Amazon AWS. Check **here** to see if this port is blocked on your machine or not. If **this page** doesn't load, your firewall is probably blocking port 27017. If it is blocked, make sure to unblock it before you proceed. How this is done will depend on the firewall you are using. Google the appropriate instruction to unblock port 27017 for the firewall you are running.

From the "Security" tab, you can also add users and manage the rights of the users you allow to access your database.

Manage users and teams

As the database administrator, you have to manage who is able to access your database and what they can do with your database. For your next MongoDB tasks, you are required to give a mentor access to your database. To do this, select "Access Manager" and then click on "Project Access", along with your current project displayed next to it on the drop-down. On the next screen, click on "Invite to Project". You can then invite a mentor to be a user of your database by entering an email address, as indicated in the image below. Please use the email address: **allreviewers@hyperiondev.com**.



Invite to Project

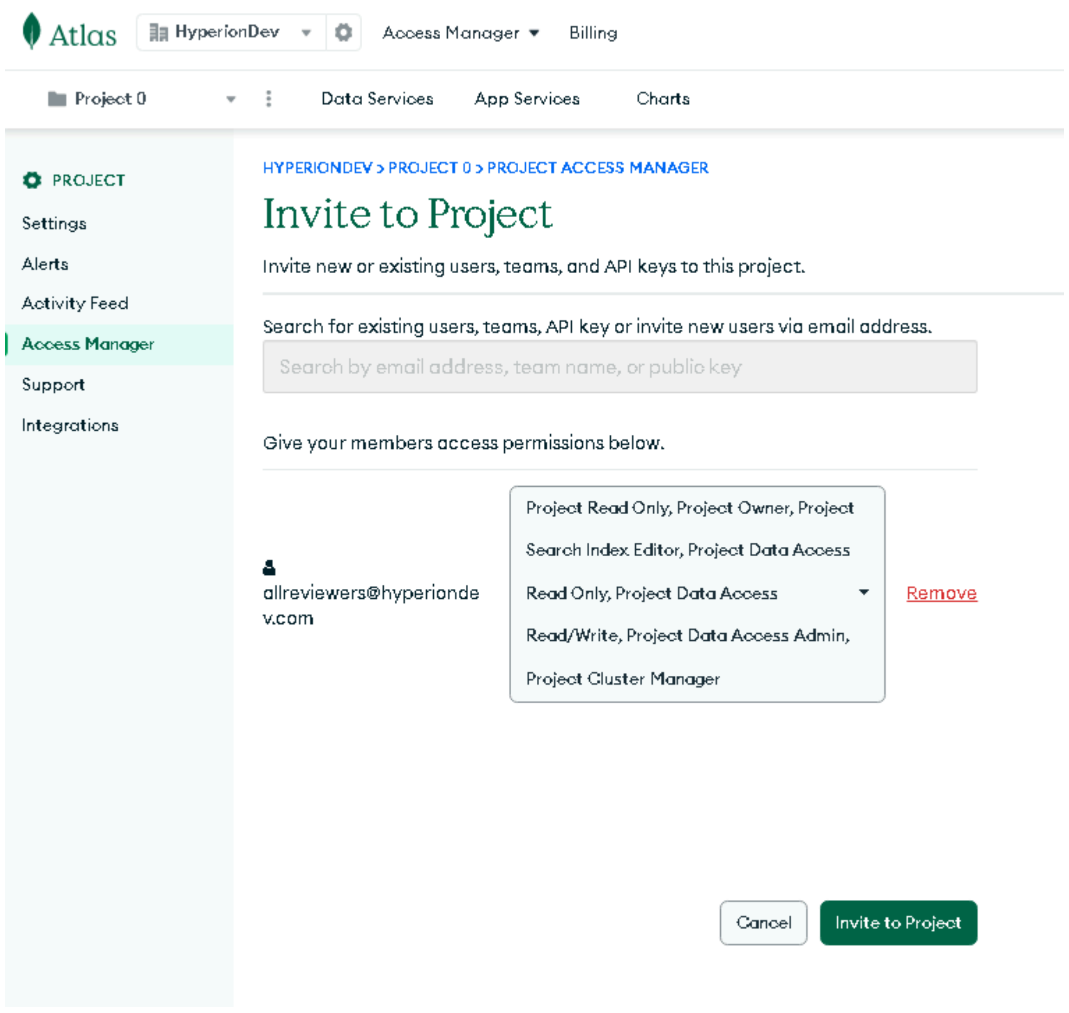
Invite new or existing users, teams, and API keys to this project.

Search for existing users, teams, API key or invite new users via email address.

allreviewers@hyperiondev.com

After pasting the email address, a suggestion should appear below it, and you'll need to click it in order to select the user.

Note: please ensure you give the code reviewer **full** access to your project.



HYPERIONDEV > PROJECT 0 > PROJECT ACCESS MANAGER

Invite to Project

Invite new or existing users, teams, and API keys to this project.

Search for existing users, teams, API key or invite new users via email address.

Search by email address, team name, or public key

Give your members access permissions below.

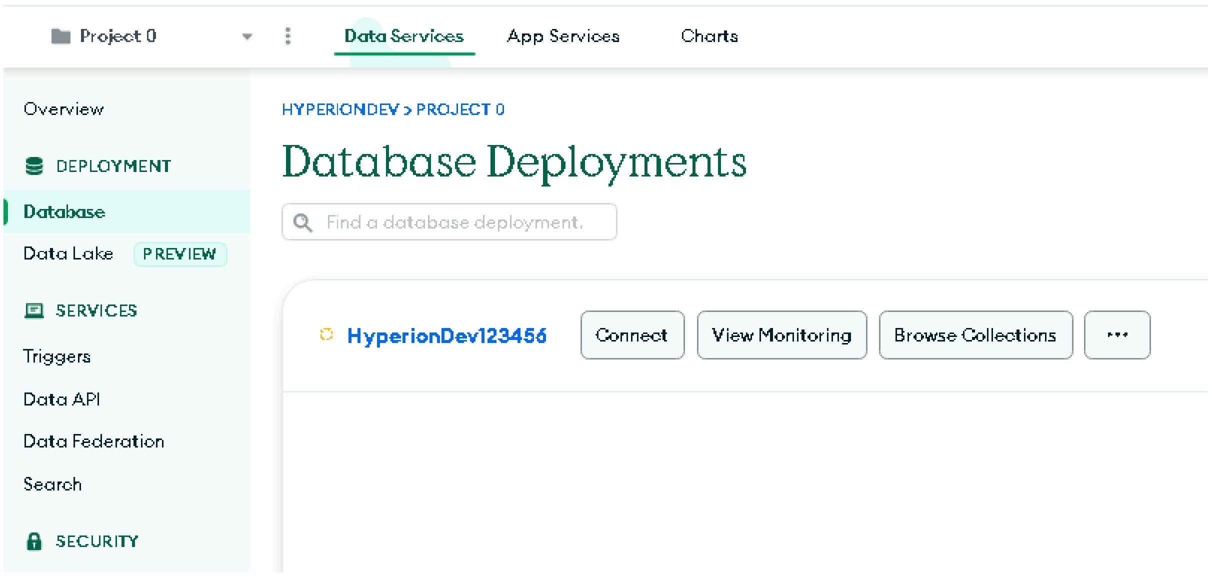
allreviewers@hyperiondev.com

Project Read Only, Project Owner, Project Search Index Editor, Project Data Access Read Only, Project Data Access Read/Write, Project Data Access Admin, Project Cluster Manager

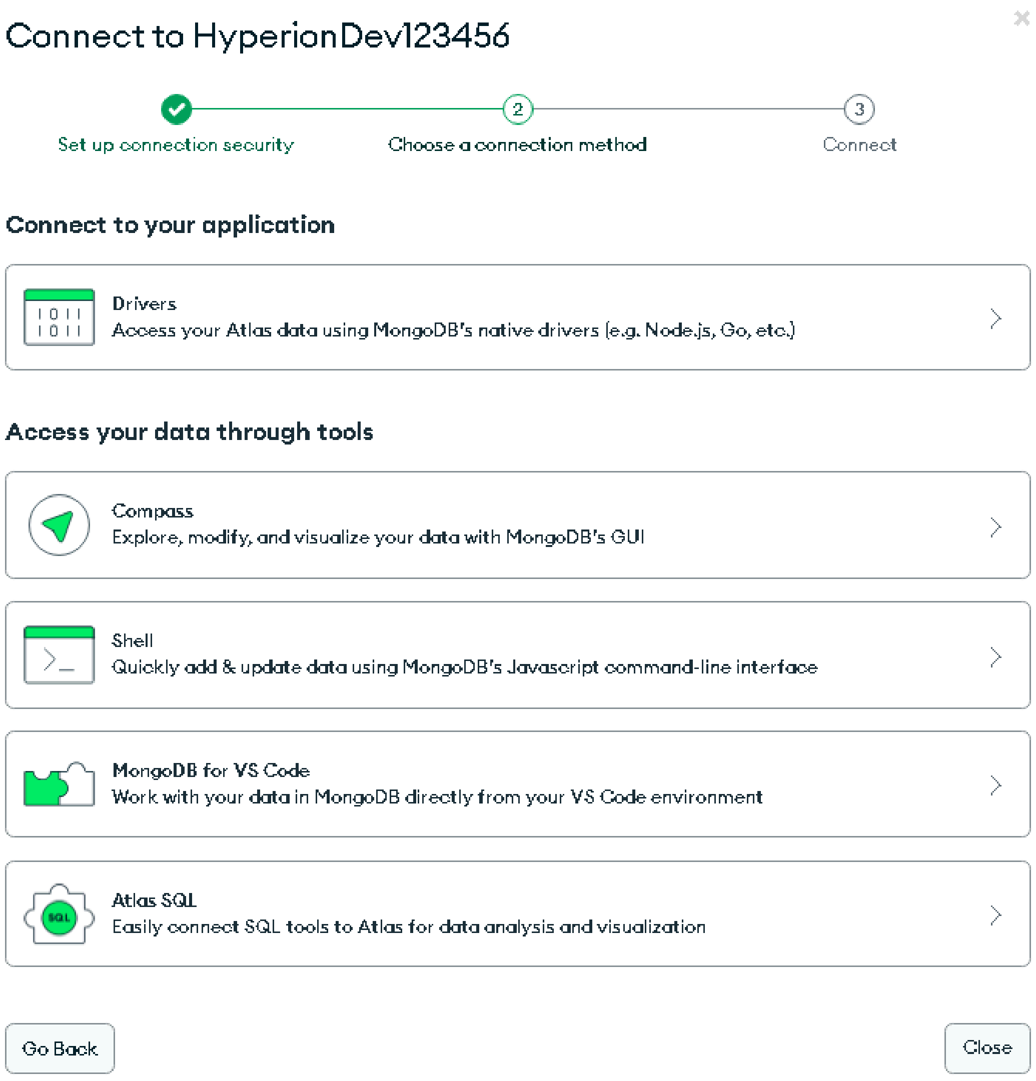
ACCESS THE DATABASE ON THE CLOUD USING THE MONGO SHELL

You are now ready to access the database server you have configured on MongoDB Atlas using the Mongo shell on your local machine. Remember that Mongo is the administrative shell used to run instructions on your MongoDB server.

To connect to the database server using Mongo, you need a connection string that specifies everything needed for this connection. Atlas provides this connection string for you. Select "Connect" as shown below, to find the connection string.



The following pop-up window will appear:


Connect to HyperionDev123456

Connect to your application

Drivers
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

Access your data through tools

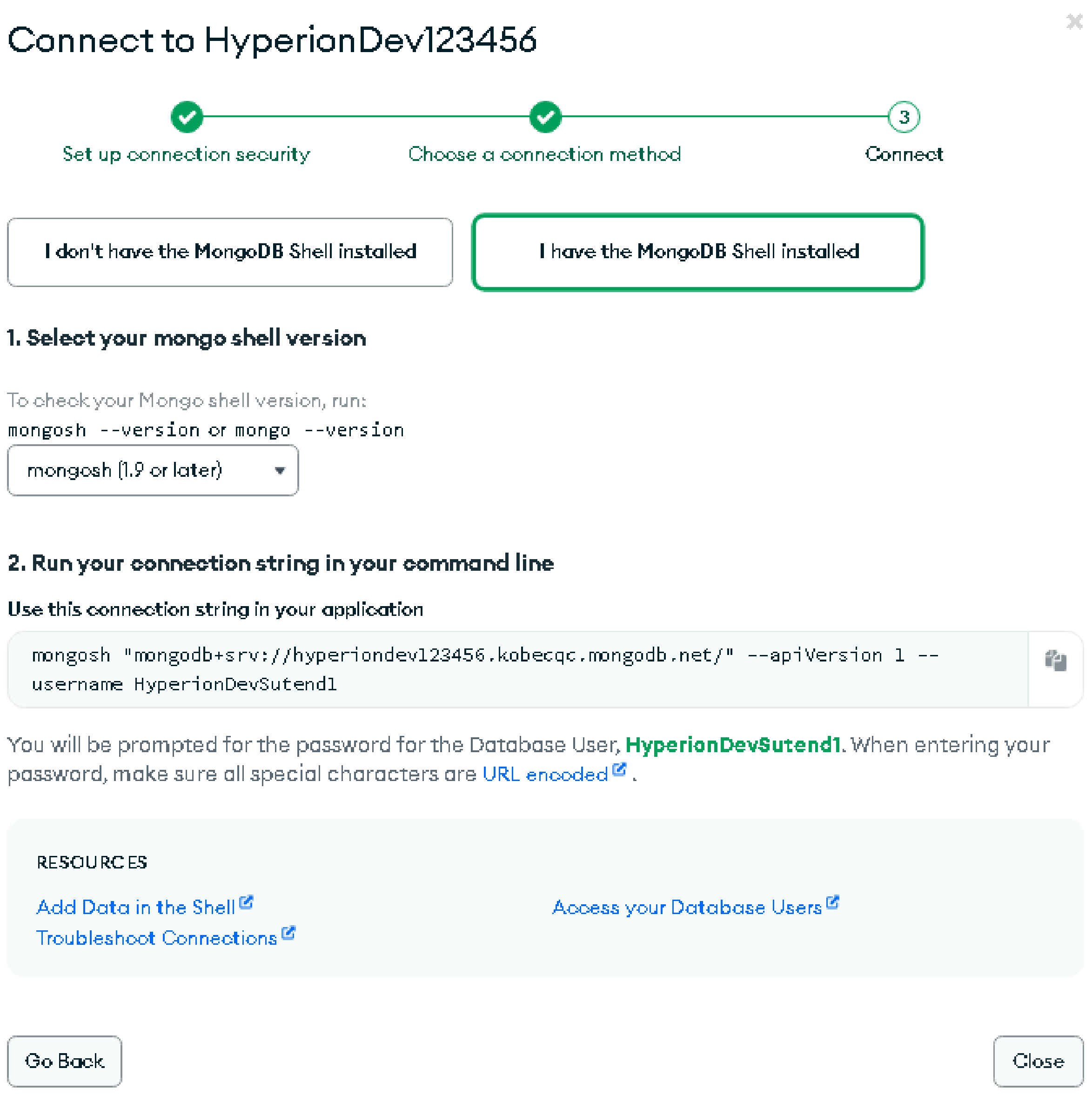
Compass
Explore, modify, and visualize your data with MongoDB's GUI

Shell
Quickly add & update data using MongoDB's Javascript command-line interface

MongoDB for VS Code
Work with your data in MongoDB directly from your VS Code environment

Atlas SQL
Easily connect SQL tools to Atlas for data analysis and visualization

From the window above, select “Shell”. The window below should then appear. Select the version of Mongo shell you have installed (for example, in our case I selected: “mongosh (1.9 or later)”, but you may need to select the version that works for you):


Connect to HyperionDev123456
Set up connection security
Choose a connection method
3
Connect
I don't have the MongoDB Shell installed
I have the MongoDB Shell installed

1. Select your mongo shell version

To check your Mongo shell version, run:
`mongosh --version` or `mongo --version`

mongosh (1.9 or later)

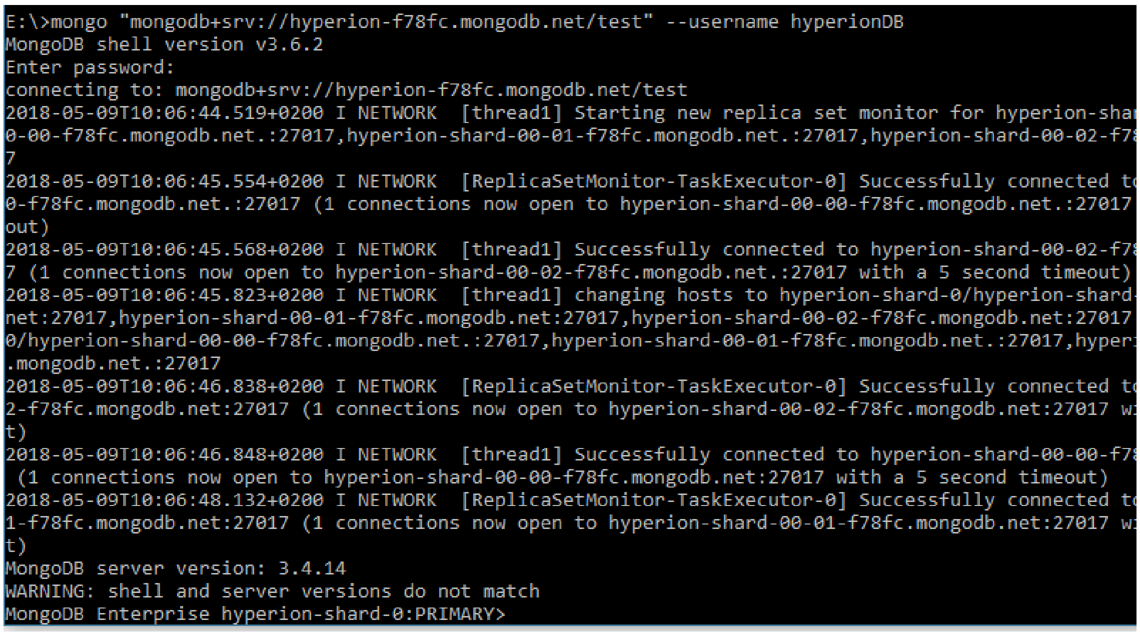
2. Run your connection string in your command line

Use this connection string in your application

```
mongosh "mongodb+srv://hyperiondev123456.kobecqc.mongodb.net/" --apiVersion 1 --
username HyperionDevSutend1
```

You will be prompted for the password for the Database User, **HyperionDevSutend1**. When entering your password, make sure all special characters are URL encoded.

This will create a new connection string. Copy and paste the connection string that appears there into your command-line interface. You should see output similar to that shown in the image below:



```
E:\>mongo "mongodb+srv://hyperion-f78fc.mongodb.net/test" --username hyperionDB
MongoDB shell version v3.6.2
Enter password:
connecting to: mongodb+srv://hyperion-f78fc.mongodb.net/test
2018-05-09T10:06:44.519+0200 I NETWORK  [thread1] Starting new replica set monitor for hyperion-sha
0-00-f78fc.mongodb.net.:27017,hyperion-shard-00-01-f78fc.mongodb.net.:27017,hyperion-shard-00-02-f7
7
2018-05-09T10:06:45.554+0200 I NETWORK  [ReplicaSetMonitor-TaskExecutor-0] Successfully connected t
0-f78fc.mongodb.net.:27017 (1 connections now open to hyperion-shard-00-00-f78fc.mongodb.net.:27017
out)
2018-05-09T10:06:45.568+0200 I NETWORK  [thread1] Successfully connected to hyperion-shard-00-02-f7
7 (1 connections now open to hyperion-shard-00-02-f78fc.mongodb.net.:27017 with a 5 second timeout)
2018-05-09T10:06:45.823+0200 I NETWORK  [thread1] changing hosts to hyperion-shard-0/hyperion-shard
net:27017,hyperion-shard-00-01-f78fc.mongodb.net:27017,hyperion-shard-00-02-f78fc.mongodb.net:27017
0/hyperion-shard-00-00-f78fc.mongodb.net.:27017,hyperion-shard-00-01-f78fc.mongodb.net.:27017,hyper
.mongodb.net.:27017
2018-05-09T10:06:46.838+0200 I NETWORK  [ReplicaSetMonitor-TaskExecutor-0] Successfully connected t
2-f78fc.mongodb.net:27017 (1 connections now open to hyperion-shard-00-02-f78fc.mongodb.net:27017 w
t)
2018-05-09T10:06:46.848+0200 I NETWORK  [thread1] Successfully connected to hyperion-shard-00-00-f7
 (1 connections now open to hyperion-shard-00-00-f78fc.mongodb.net:27017 with a 5 second timeout)
2018-05-09T10:06:48.132+0200 I NETWORK  [ReplicaSetMonitor-TaskExecutor-0] Successfully connected t
1-f78fc.mongodb.net:27017 (1 connections now open to hyperion-shard-00-01-f78fc.mongodb.net:27017 w
t)
MongoDB server version: 3.4.14
WARNING: shell and server versions do not match
MongoDB Enterprise hyperion-shard-0:PRIMARY>
```

You have connected to your MongoDB server hosted by Atlas! You are now able to use the Mongo shell to create and modify databases on your server.

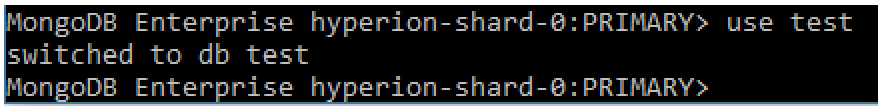
Extra resource

Watch **this short video** to see how easily you can connect to your Atlas database to the Mongo shell.

CREATE A DATABASE

Once you can access your database server (run by Atlas), you can issue instructions using Mongo to change your database. We are going to create a database.

To do this, type the following using the Mongo shell: `use test` (where 'test' is the name of the database). If the database does not already exist, this instruction will create it.



```
MongoDB Enterprise hyperion-shard-0:PRIMARY> use test
switched to db test
MongoDB Enterprise hyperion-shard-0:PRIMARY>
```

MONGODB COMPASS

You may have noticed that when you installed MongoDB, **MongoDB Compass** was also installed. Compass allows you to interface with your database. You should be able to connect to your database using Compass, too. Give it a try.

QUIT MONGO

To quit mongo, type `quit()` into the Mongo shell.

Take note

The tasks below are **auto-graded**. An auto-graded task still counts towards your progression and graduation. Give the task your best attempt and submit it when you are ready.

After you click "Request review" on your student dashboard, you will receive a 100% pass grade if you've submitted the task.

When you submit the task, you will receive an email with a link to a model answer, as well as an overview of the approach taken to reach this answer. Take some time to review and compare your work against the model answer. This exercise will help solidify your understanding and provide an opportunity for reflection on how to apply these concepts in future projects.

In the same email, you will also receive a link to a survey for this task, which you can use as a self-assessment tool. Please take a moment to complete the survey.

Once you've done that, feel free to progress to the next task.

Instructions

Take your time and work carefully through the practical task instructions below, one point at a time.

Auto-graded task

Follow these steps:

- Install MongoDB. See appropriate detailed installation instructions on MongoDB's download centre for details.
- Create a cluster on MongoDB Atlas.
- Add a mentor or code reviewer as a user to your cluster on Atlas, using the email address **allreviewers@hyperiondev.com**.
- Ensure that your firewall isn't blocking access to MongoDB.
- Connect to your cluster using the Mongo shell.
- Make a database called **test**.
- Create a document called **myMongoDB**, in which you include the following:
    - A screenshot that shows that you have added a mentor or code reviewer as a MongoDB user to your Atlas cluster.
    - A screenshot of your command-line interface that shows how you have used the Mongo shell to connect to your MongoDB Atlas cluster.
    - A screenshot of your command-line interface that shows that you have successfully created a database called 'test' on your MongoDB Atlas cluster.

Be sure to place files for submission inside your task folder and click "Request review" on your dashboard.

Rate us

Share your thoughts

HyperionDev strives to provide internationally excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved, or think we’ve done a good job?

Click here to share your thoughts anonymously.