

# Node Driver Quick Start

## Overview

This guide shows you how to create an application that uses the MongoDB Node.js driver to connect to a MongoDB cluster hosted on MongoDB Atlas. If you prefer to connect to MongoDB using a different driver or programming language, see our [list of official drivers](#).

The Node.js driver is a library of functions that you can use to connect to and communicate with MongoDB.

MongoDB Atlas is a fully managed cloud database service that hosts your MongoDB deployments. You can create your own free (no credit card required) MongoDB Atlas deployment by following the steps in this guide.

Follow the steps in this guide to connect a sample Node.js application to a MongoDB Atlas deployment.

Next: Download and Install

← [MongoDB Node Driver](#)

[Download and Install](#) →

On this page

Overview

★ Rate this page

✦ Ask MongoDB AI



# Download and Install

1

## Install Node and npm

Ensure you have Node.js v16 or later and npm (Node Package Manager) installed in your development environment.

For information on how to install Node.js and npm, see [downloading and installing Node.js and npm](#).

2

## Create a Project Directory

In your shell, run the following command to create a directory called `node_quickstart` for this project:

```
mkdir node_quickstart
```

Run the following command to navigate into the project directory:

```
cd node_quickstart
```

Run the following command to initialize your Node.js project:

```
npm init -y
```

When this command successfully completes, you have a `package.json` file in your `node_quickstart` directory.

3

## Install the Node.js Driver

Run the following command in your shell to install the driver in your project directory:

```
npm install mongodb@6.6
```

This command performs the following actions:

- Downloads the `mongodb` package and the dependencies it requires
- Saves the package in the `node_modules` directory
- Records the dependency information in the `package.json` file

After you complete these steps, you have Node.js and npm installed and a new project directory with the driver dependencies installed.

### NOTE

If you run into issues on this step, ask for help in the [MongoDB Community Forums](#) or submit feedback by using the **Rate this page** tab on the right or bottom right side of this page.

Next: Create a MongoDB Deployment

★ Rate this page

🌟 Ask MongoDB AI

# Create a MongoDB Deployment

You can create a free tier MongoDB deployment on MongoDB Atlas to store and manage your data. MongoDB Atlas hosts and manages your MongoDB database in the cloud.

- 1

## Create a Free MongoDB deployment on Atlas

Complete the [Get Started with Atlas](#) guide to set up a new Atlas account and load sample data into a new free tier MongoDB deployment.
- 2

## Save your Credentials

After you create your database user, save that user's username and password to a safe location for use in an upcoming step.

After you complete these steps, you have a new free tier MongoDB deployment on Atlas, database user credentials, and sample data loaded in your database.

**NOTE**

If you run into issues on this step, ask for help in the [MongoDB Community Forums](#) or submit feedback by using the **Rate this page** tab on the right or bottom right side of this page.



Next: Create a Connection String

## Create a Connection String

You can connect to your MongoDB deployment by providing a **connection URI**, also called a *connection string*, which instructs the driver on how to connect to a MongoDB deployment and how to behave while connected.

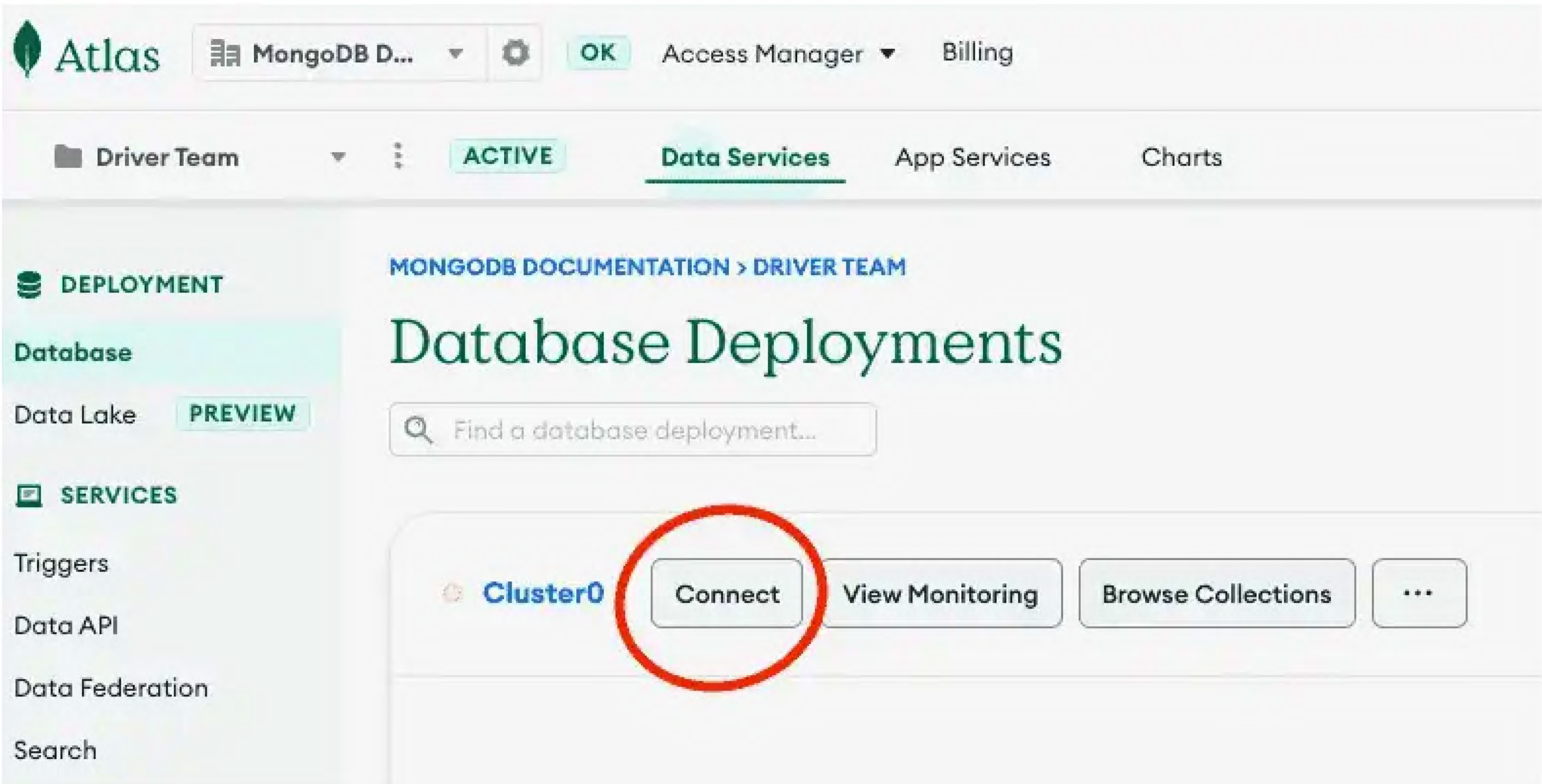
The connection string includes the hostname or IP address and port of your deployment, the authentication mechanism, user credentials when applicable, and connection options.

To connect to an instance or deployment not hosted on Atlas, see [Other Ways to Connect to MongoDB](#).

1

### Find your MongoDB Atlas Connection String

To retrieve your connection string for the deployment that you created in the [previous step](#), log into your Atlas account and navigate to the **Database** section and click the **Connect** button for your new deployment.



★ Rate this page

✦ Ask MongoDB AI

Proceed to the **Connect your application** section and select "Node.js" from the **Driver** selection menu and the version that best matches the version you installed from the **Version** selection menu.

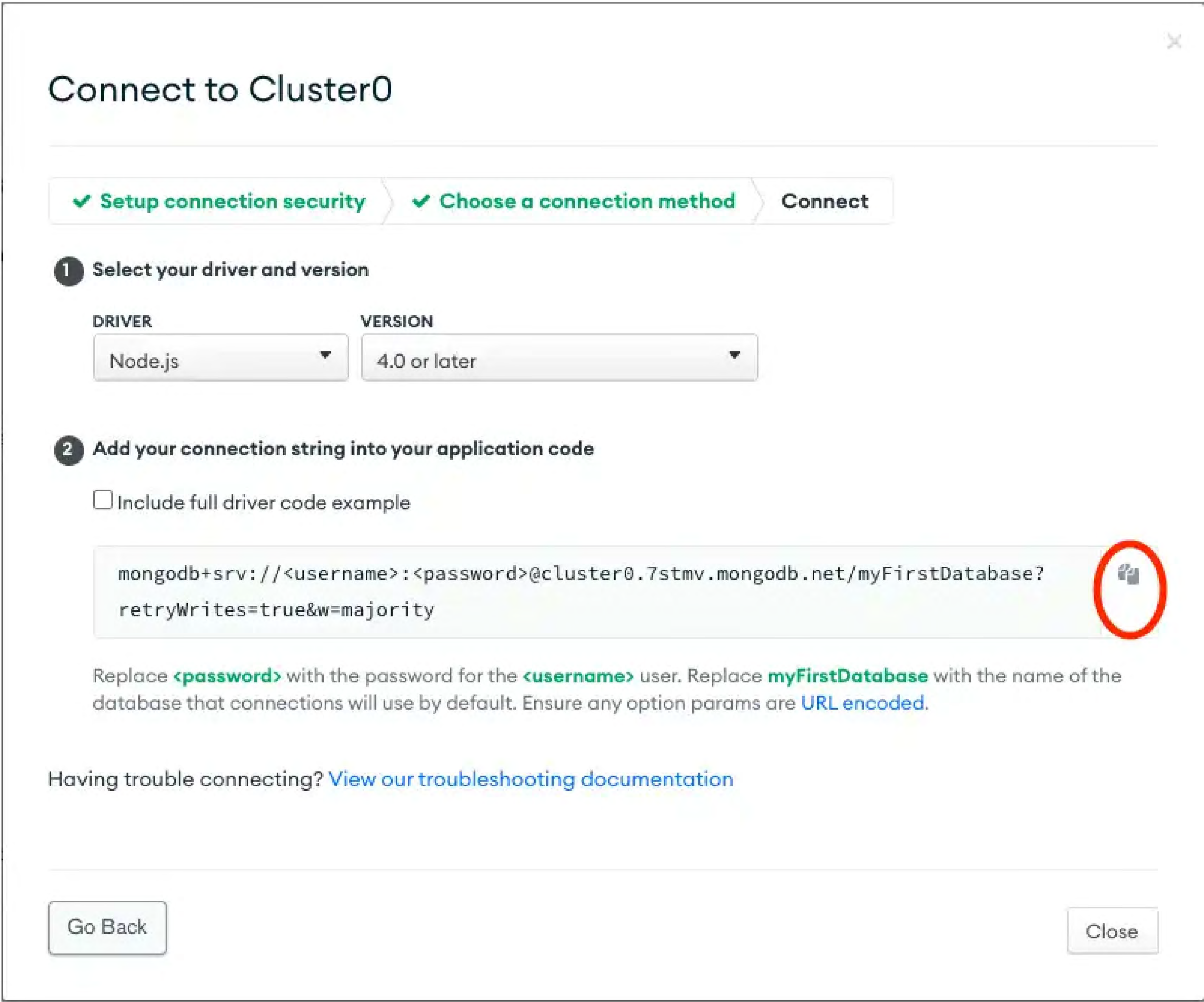
Select the **Password (SCRAM)** authentication mechanism.

Deselect the **Include full driver code example** to view the connection string.

2

### Copy your Connection String

Click the button on the right of the connection string to copy it to your clipboard as shown in the following screenshot:



3

### Update the Placeholders

Paste this connection string into a a file in your preferred text editor and replace the "<username>" and "<password>" placeholders with your database user's username and password.

Save this file to a safe location for use in the next step.

After completing these steps, you have a connection string that contains your database username and password.

#### NOTE

If you run into issues on this step, ask for help in the [MongoDB Community Forums](#) or submit feedback by using the **Rate this page** tab on the right or bottom right side of this page.

Next: Connect to MongoDB

# Connect to MongoDB

1

## Create your Node.js Application

Create a file to contain your application called `index.js` in your `node_quickstart` project directory.

Copy and paste the following code into the `index.js` file:

```
const { MongoClient } = require("mongodb");

// Replace the uri string with your connection string.
const uri = "<connection string uri>";

const client = new MongoClient(uri);

async function run() {
  try {
    const database = client.db('sample_mflix');
    const movies = database.collection('movies');

    // Query for a movie that has the title 'Back to the Future'
    const query = { title: 'Back to the Future' };
    const movie = await movies.findOne(query);

    console.log(movie);
  } finally {
    // Ensures that the client will close when you finish/error
    await client.close();
  }
}
run().catch(console.dir);
```

★ Rate this page

🌟 Ask MongoDB AI

2

## Assign the Connection String

Replace the `<connection string uri>` placeholder with the connection string that you copied from the [Create a Connection String](#) step of this guide.

3

## Run your Node.js Application

In your shell, run the following command to start this application:

```
node index.js
```

The output includes details of the retrieved movie document:

```
{
  _id: ...,
  plot: 'A young man is accidentally sent 30 years into the past...',
  genres: [ 'Adventure', 'Comedy', 'Sci-Fi' ],
  ...
  title: 'Back to the Future',
  ...
}
```

If you encounter an error or see no output, check whether you specified the proper connection string in the `index.js` file, and that you loaded the sample data.

After you complete these steps, you have a working application that uses the driver to connect to your MongoDB deployment, runs a query on the sample data, and prints out the result.

### NOTE

If you run into issues on this step, ask for help in the [MongoDB Community Forums](#) or submit feedback by using the **Rate this page** tab on the right or bottom right side of this page.

Next Steps

## Next Steps

Congratulations on completing the quick start tutorial!

In this tutorial, you created a Node.js application that connects to a MongoDB deployment hosted on MongoDB Atlas and retrieves a document that matches a query.

Learn more about the MongoDB Node.js driver from the following resources:

- Discover how to perform read and write operations in the [CRUD Operations](#) section.
- See examples of frequently-used operations in the [Usage Examples](#) section.

← [Connect to MongoDB](#)

[Quick Reference](#) →

★ [Rate this page](#)

✦ [Ask MongoDB AI](#)

