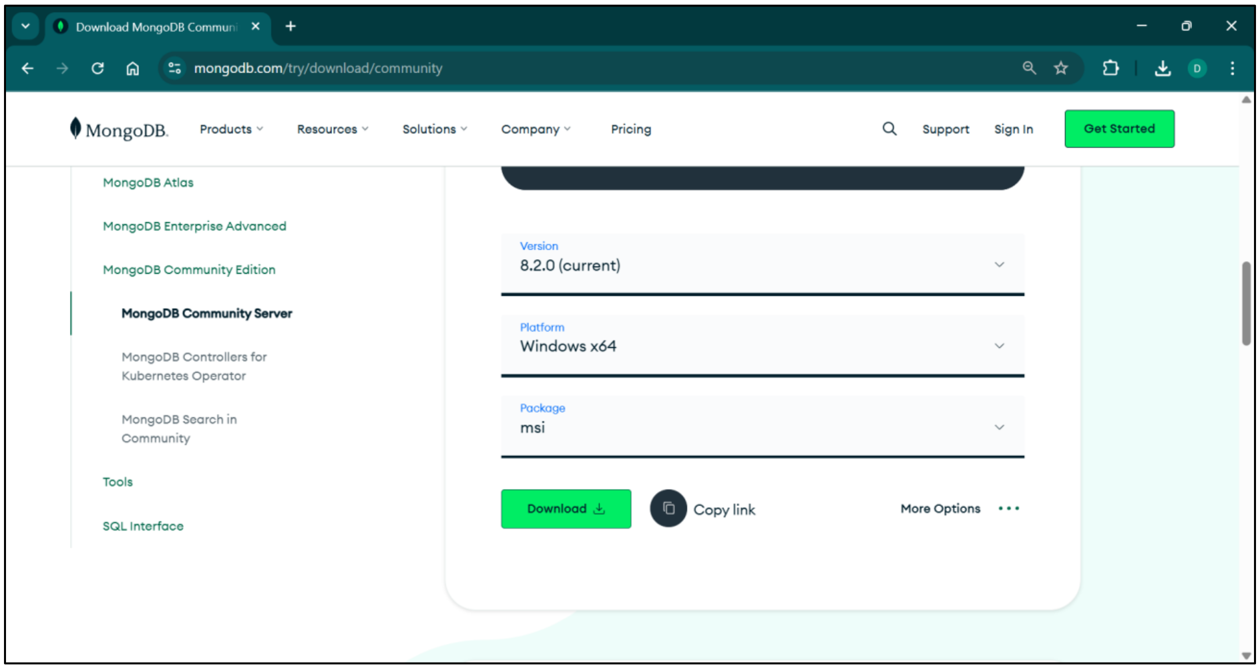
11. Installing MongoDB. Illustrate the following – inserting, finding and querying data, importing data.

Download MongoDB

Go to the MongoDB Community Server download page
https://www.mongodb.com/try/download/community

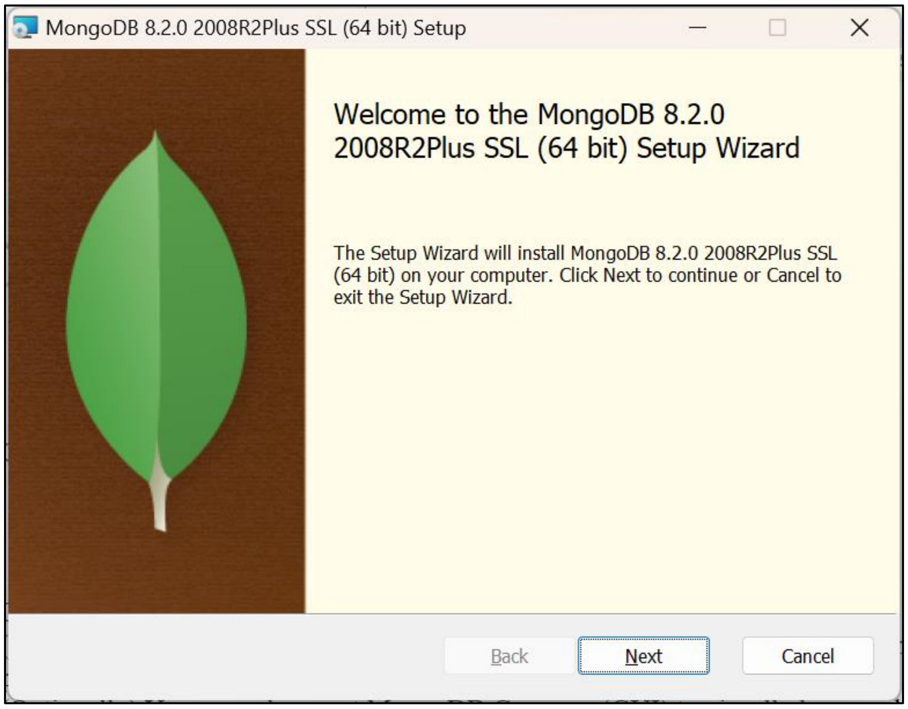


Version: 8.2.0 (current)
Platform: Windows x64
Package: msi

Select the **version** 8.2.0 (current), **OS / platform** (Windows x64) and the package type (msi).

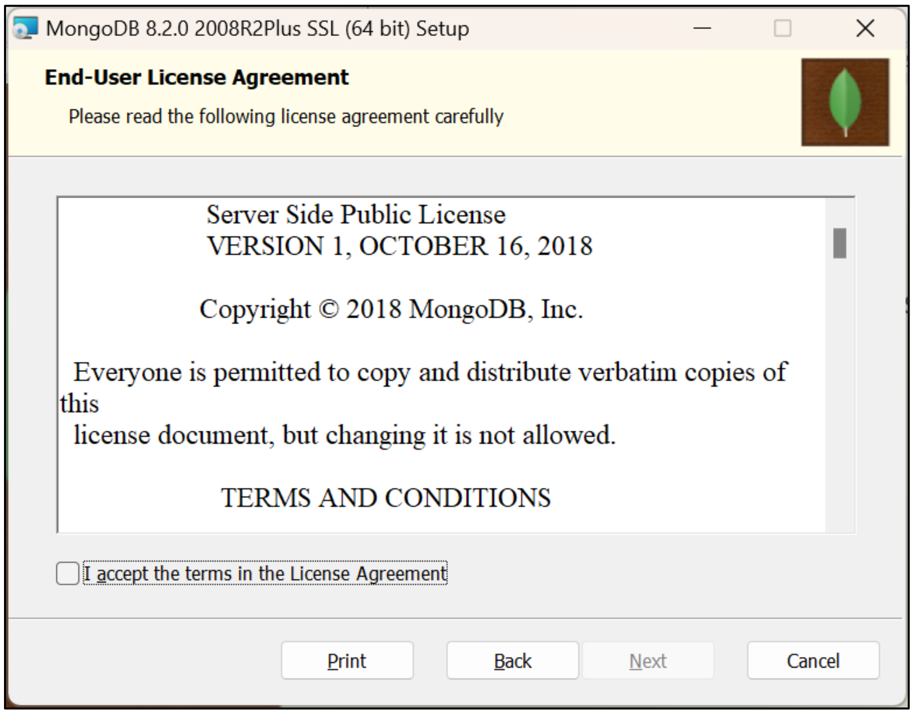
Click on **Download**.

Install MongoDB


MongoDB 8.2.0 2008R2Plus SSL (64 bit) Setup

Welcome to the MongoDB 8.2.0 2008R2Plus SSL (64 bit) Setup Wizard

The Setup Wizard will install MongoDB 8.2.0 2008R2Plus SSL (64 bit) on your computer. Click Next to continue or Cancel to exit the Setup Wizard.


MongoDB 8.2.0 2008R2Plus SSL (64 bit) Setup

End-User License Agreement

Please read the following license agreement carefully

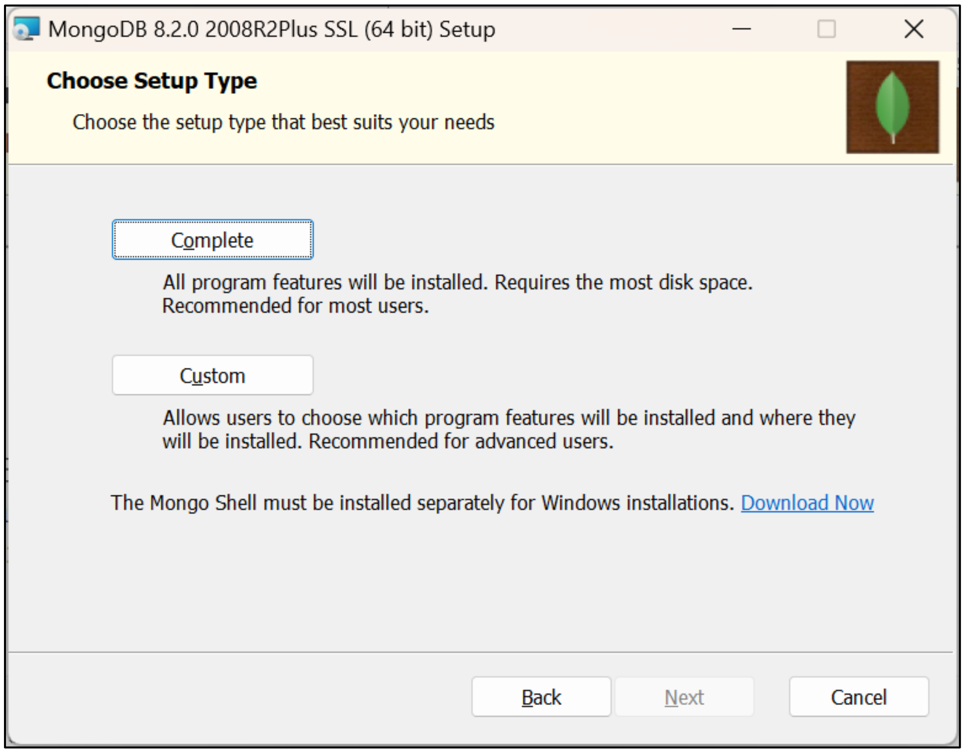
Server Side Public License
VERSION 1, OCTOBER 16, 2018

Copyright © 2018 MongoDB, Inc.

Everyone is permitted to copy and distribute verbatim copies of this
license document, but changing it is not allowed.

TERMS AND CONDITIONS

☐ I accept the terms in the License Agreement


MongoDB 8.2.0 2008R2Plus SSL (64 bit) Setup

Choose Setup Type

Choose the setup type that best suits your needs

Complete

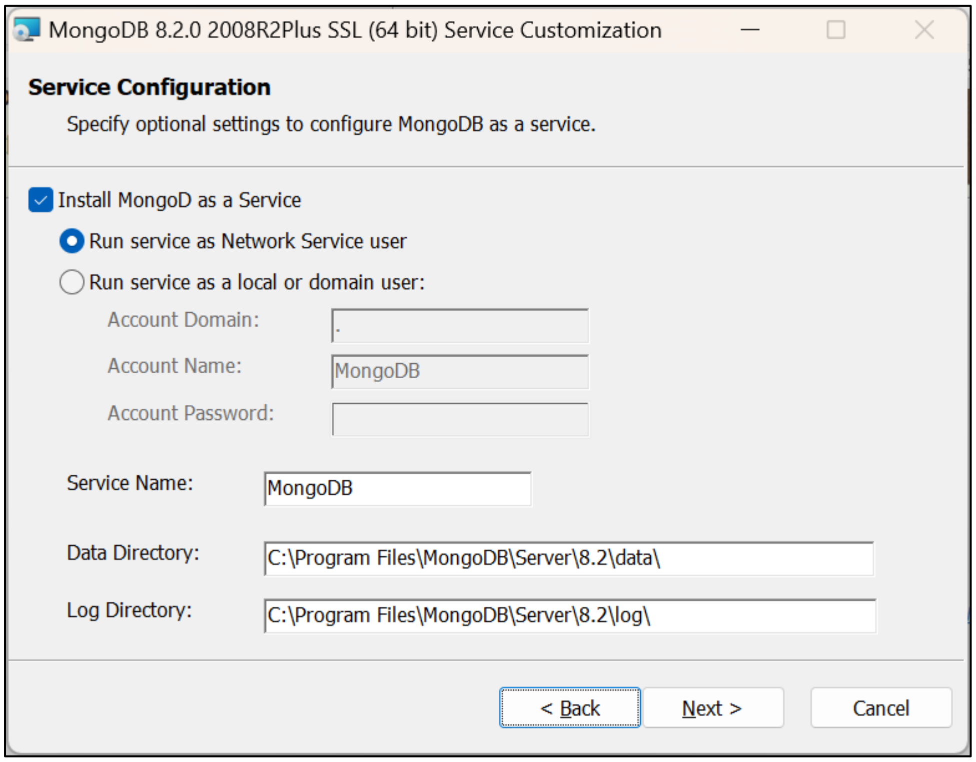
All program features will be installed. Requires the most disk space. Recommended for most users.

Custom

Allows users to choose which program features will be installed and where they will be installed. Recommended for advanced users.

The Mongo Shell must be installed separately for Windows installations. Download Now

Back Next Cancel


MongoDB 8.2.0 2008R2Plus SSL (64 bit) Service Customization

Service Configuration

Specify optional settings to configure MongoDB as a service.

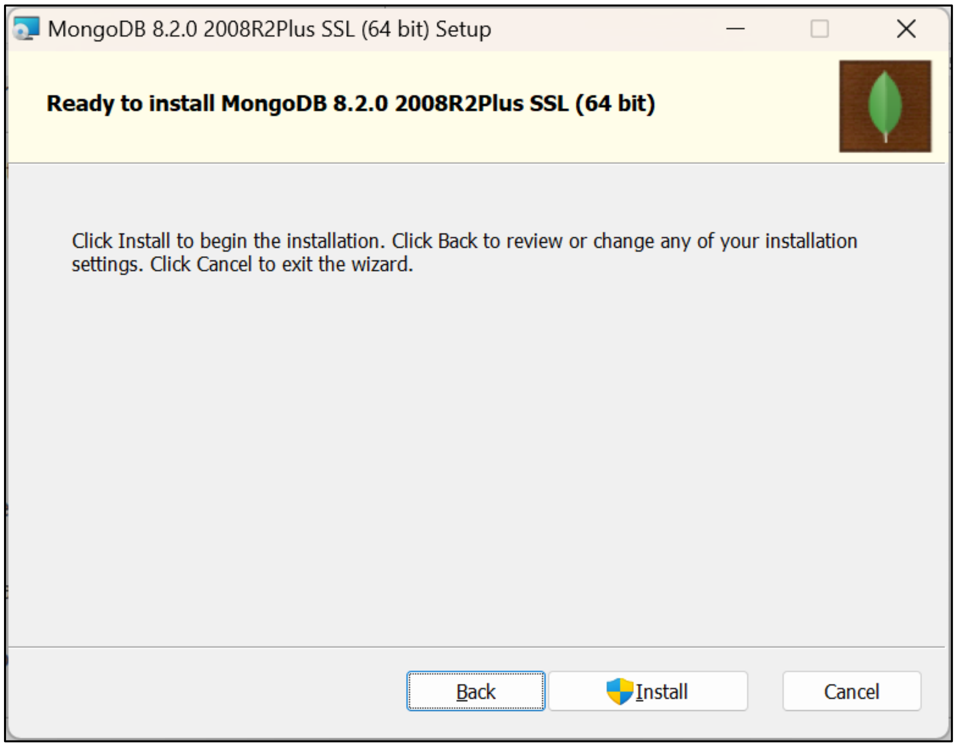
[✓] Install MongoD as a Service

[● Run service as Network Service user]

○ Run service as a local or domain user:

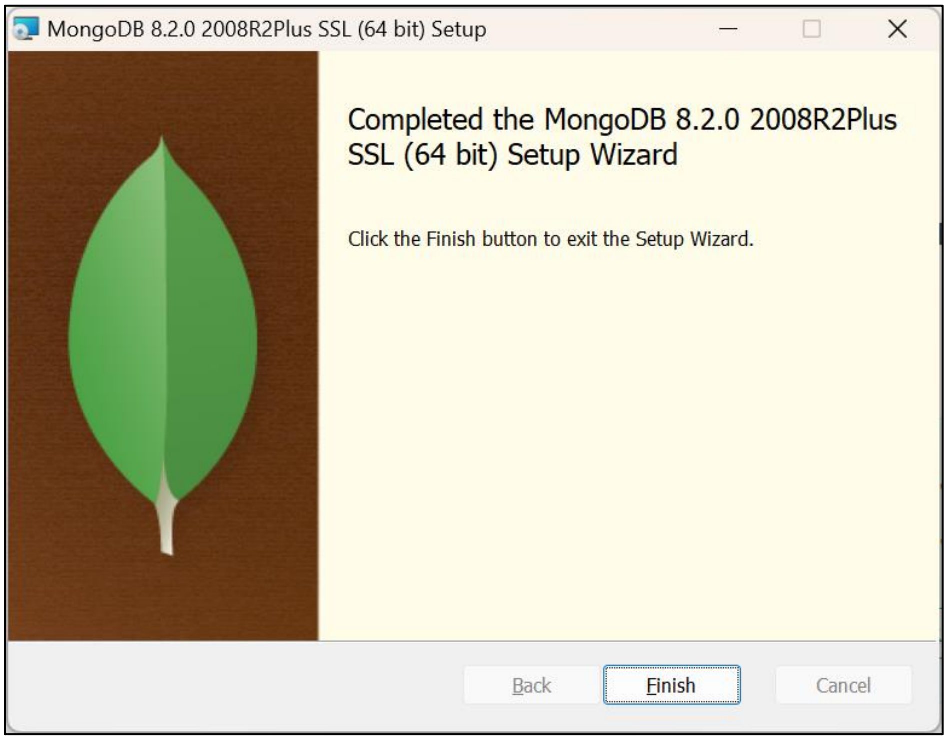
Account Domain:	.
Account Name:	MongoDB
Account Password:	
Service Name:	MongoDB
Data Directory:	C:\Program Files\MongoDB\Server\8.2\data\
Log Directory:	C:\Program Files\MongoDB\Server\8.2\log\

< Back Next > Cancel


MongoDB 8.2.0 2008R2Plus SSL (64 bit) Setup

Ready to install MongoDB 8.2.0 2008R2Plus SSL (64 bit)

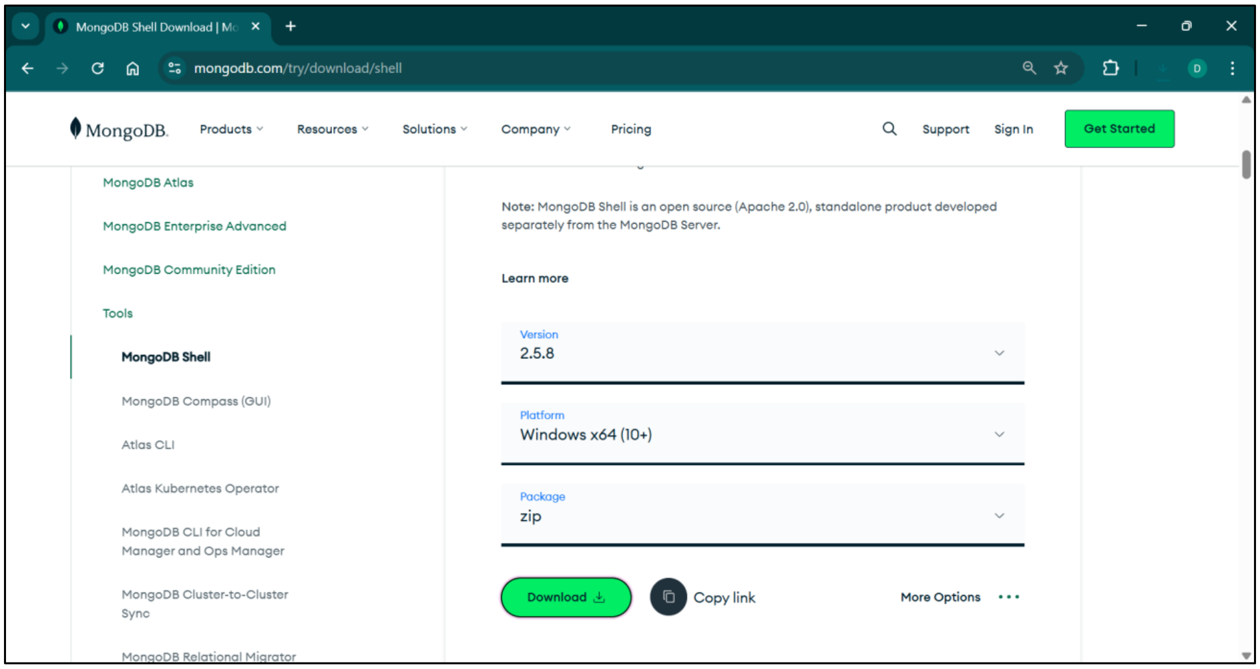
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.


MongoDB 8.2.0 2008R2Plus SSL (64 bit) Setup

Completed the MongoDB 8.2.0 2008R2Plus SSL (64 bit) Setup Wizard

Click the Finish button to exit the Setup Wizard.

Download and Install mongo shell from https://www.mongodb.com/try/download/shell

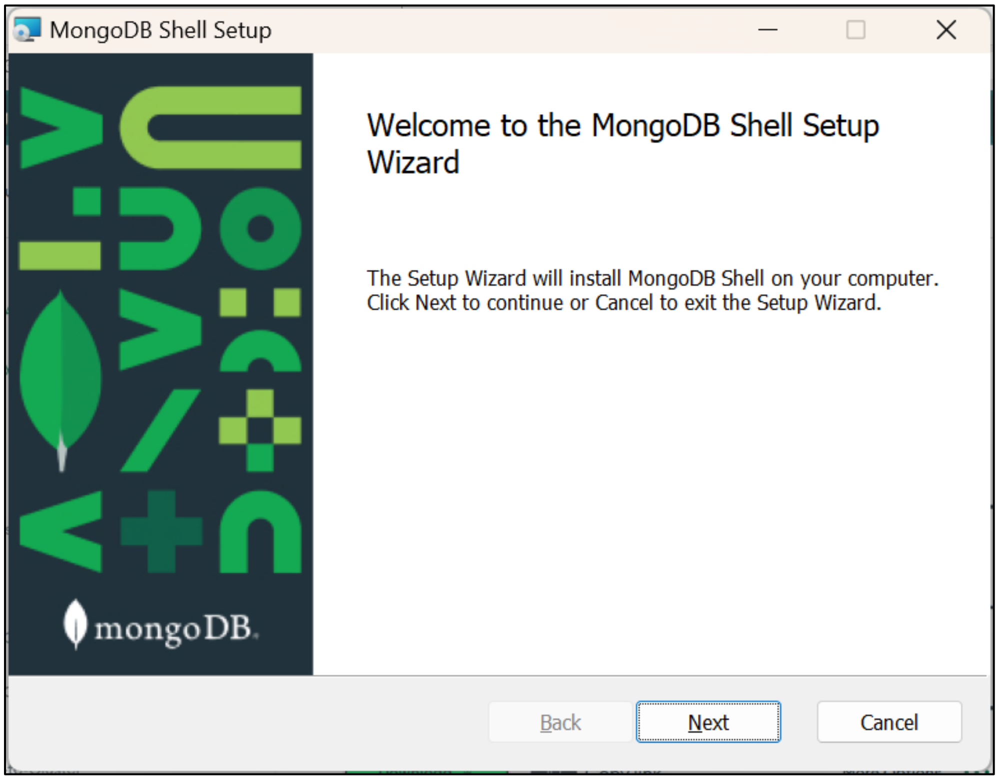

MongoDB Shell Download | Mo
mongodb.com/try/download/shell
MongoDB.
Products
Resources
Solutions
Company
Pricing
Support
Sign In
Get Started
MongoDB Atlas
MongoDB Enterprise Advanced
MongoDB Community Edition
Tools
MongoDB Shell
MongoDB Compass (GUI)
Atlas CLI
Atlas Kubernetes Operator
MongoDB CLI for Cloud Manager and Ops Manager
MongoDB Cluster-to-Cluster Sync
MongoDB Relational Migrator
Learn more
Download
Copy link
More Options

Note: MongoDB Shell is an open source (Apache 2.0), standalone product developed separately from the MongoDB Server.

Version: 2.5.8

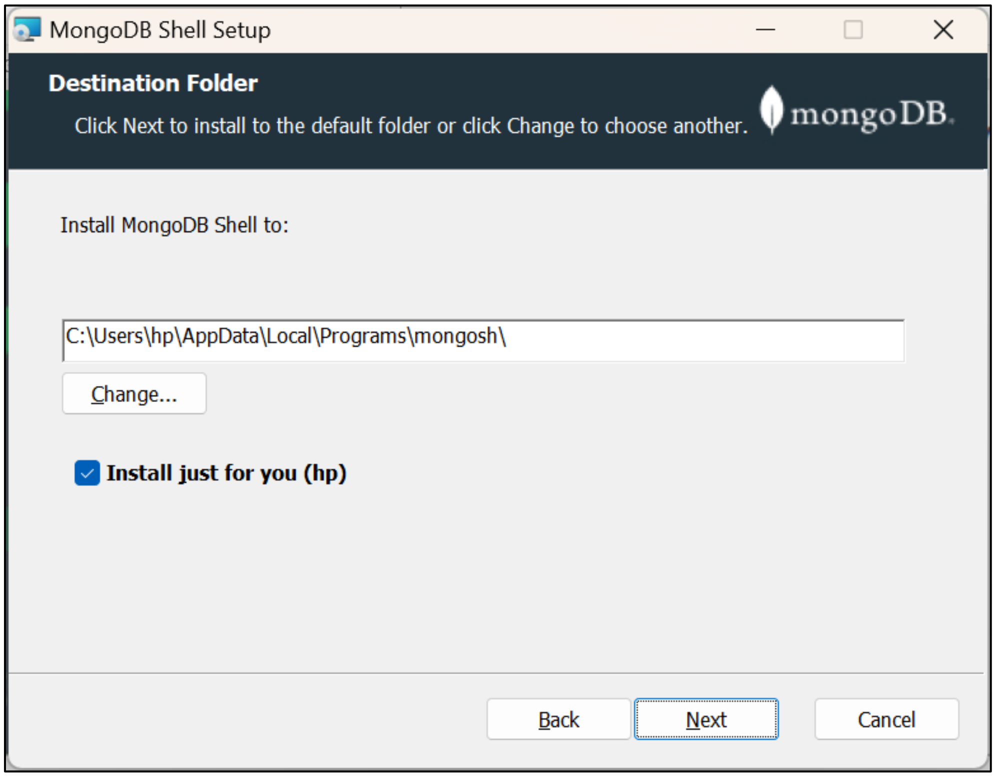
Platform: Windows x64 (10+)

Package: zip



Welcome to the MongoDB Shell Setup Wizard

The Setup Wizard will install MongoDB Shell on your computer. Click Next to continue or Cancel to exit the Setup Wizard.



MongoDB Shell Setup

Destination Folder

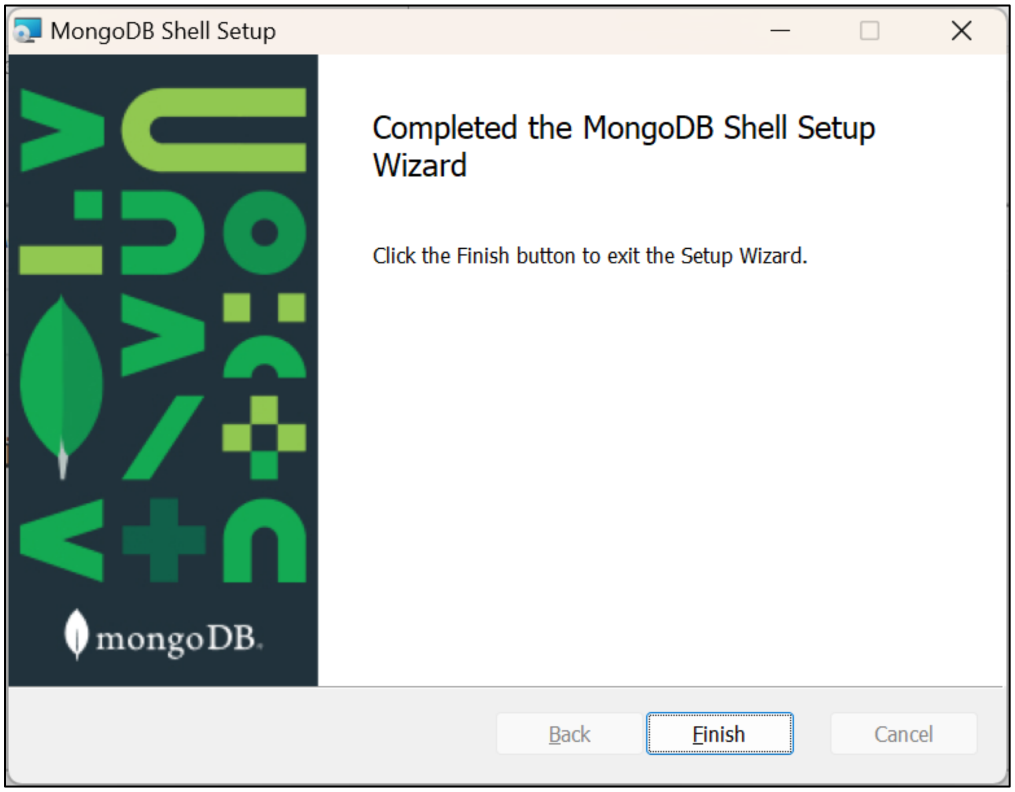
Click Next to install to the default folder or click Change to choose another.

Install MongoDB Shell to:

C:\Users\hp\AppData\Local\Programs\mongosh\

☑ **Install just for you (hp)**

Click on Next and Install.



MongoDB Shell Setup

Completed the MongoDB Shell Setup Wizard

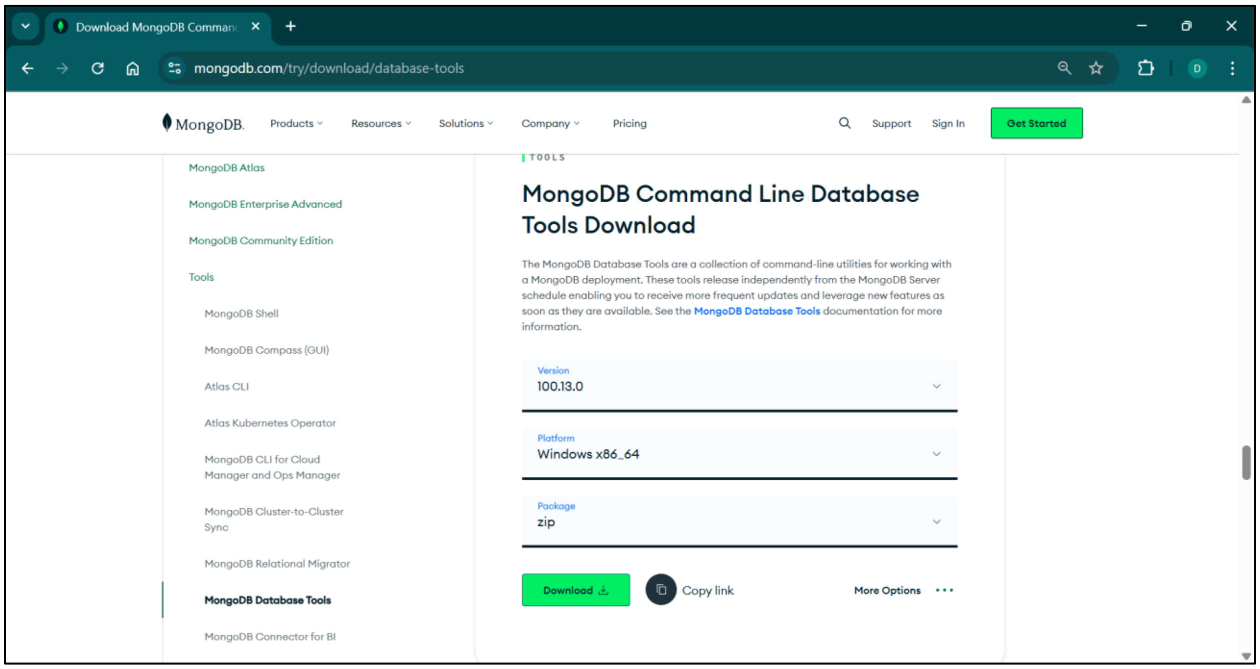
Click the Finish button to exit the Setup Wizard.

Click on Finish.

Install Database Tools separately (if missing)

If mongoimport.exe is **not in bin folder**, download **MongoDB Database Tools**:

https://www.mongodb.com/try/download/database-tools



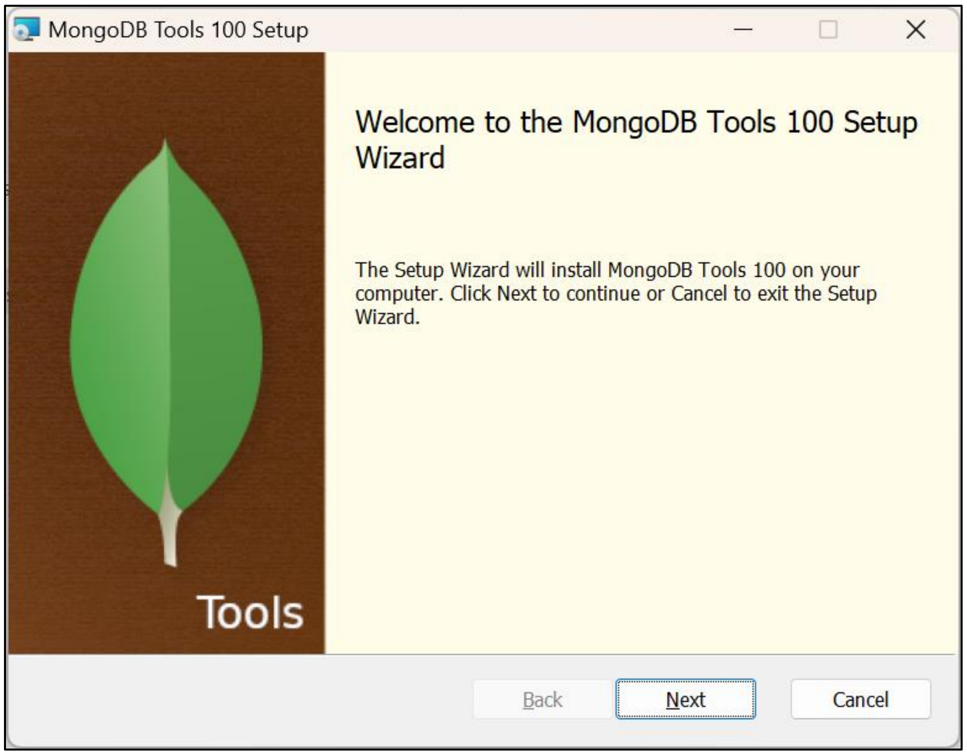
MongoDB Command Line Database Tools Download

The MongoDB Database Tools are a collection of command-line utilities for working with a MongoDB deployment. These tools release independently from the MongoDB Server schedule enabling you to receive more frequent updates and leverage new features as soon as they are available. See the MongoDB Database Tools documentation for more information.

Version: 100.13.0

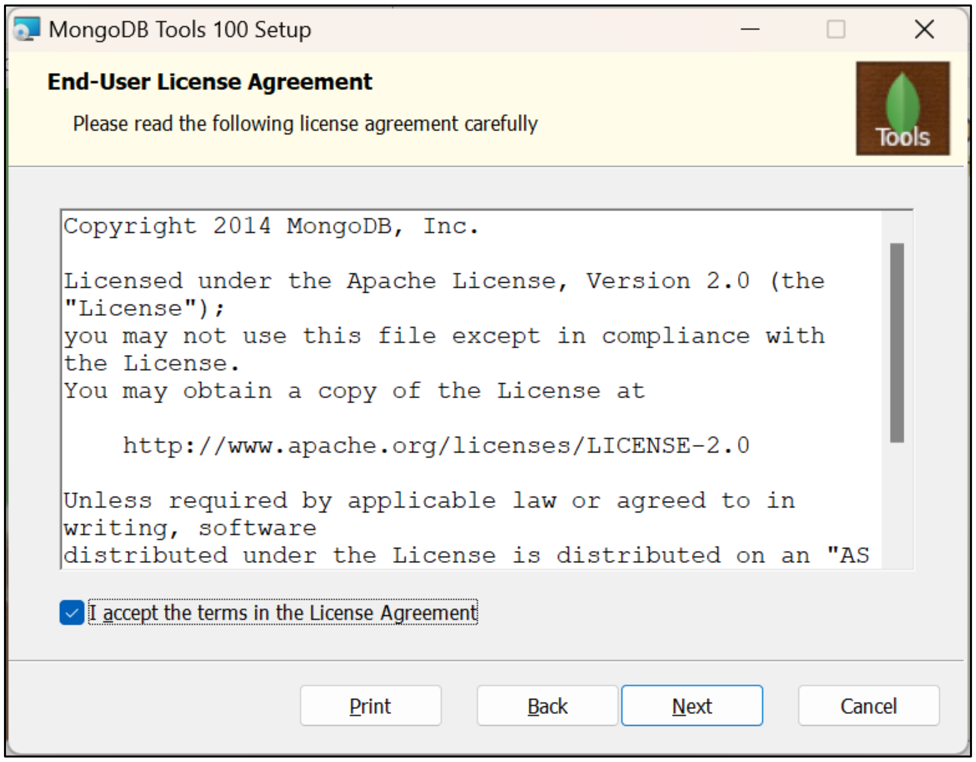
Platform: Windows x86_64

Package: zip



Welcome to the MongoDB Tools 100 Setup Wizard

The Setup Wizard will install MongoDB Tools 100 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.



End-User License Agreement

Please read the following license agreement carefully

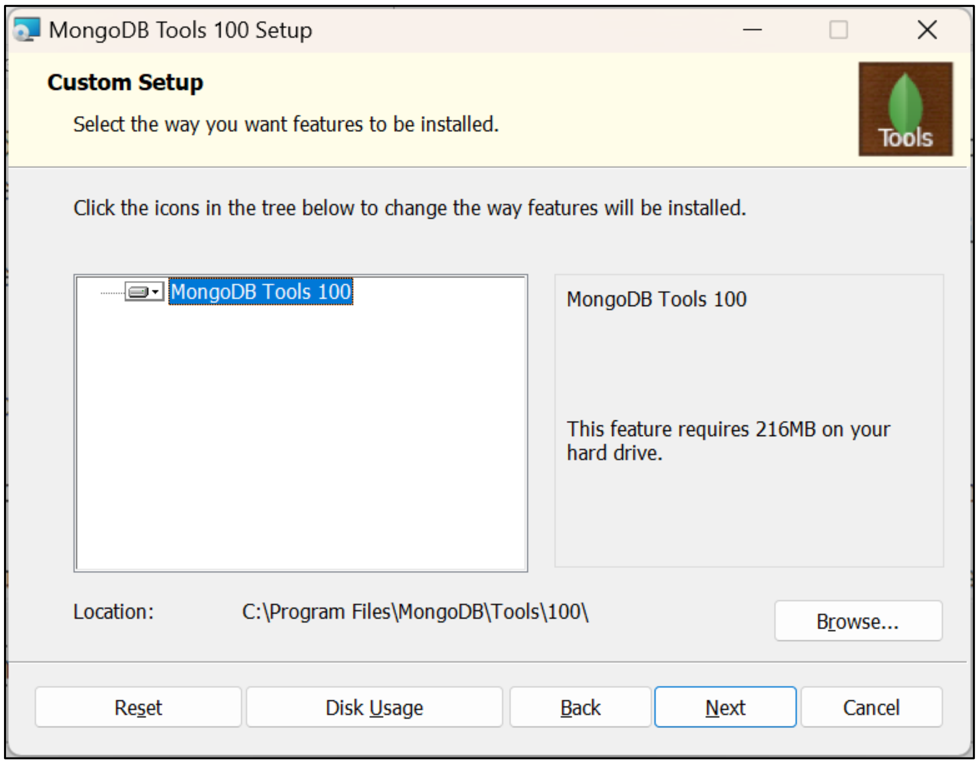
```
Copyright 2014 MongoDB, Inc.

Licensed under the Apache License, Version 2.0 (the
"License");
you may not use this file except in compliance with
the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in
writing, software
distributed under the License is distributed on an "AS
```

[✓] I accept the terms in the License Agreement



Custom Setup

Select the way you want features to be installed.

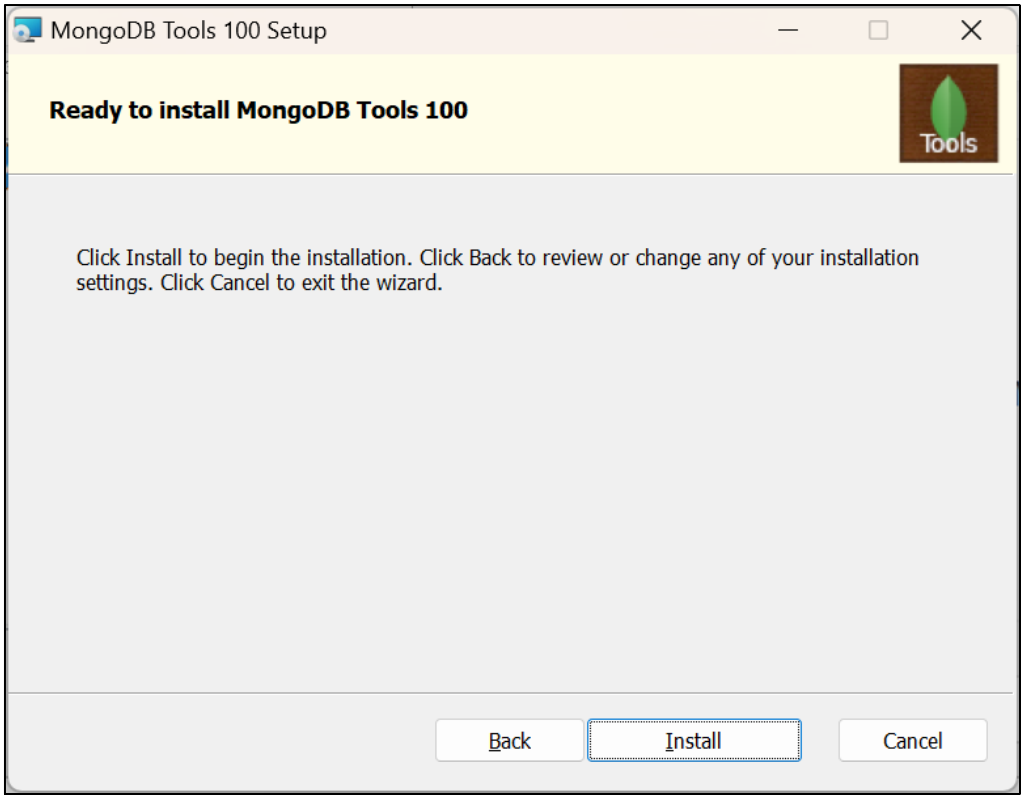
Click the icons in the tree below to change the way features will be installed.

MongoDB Tools 100

MongoDB Tools 100

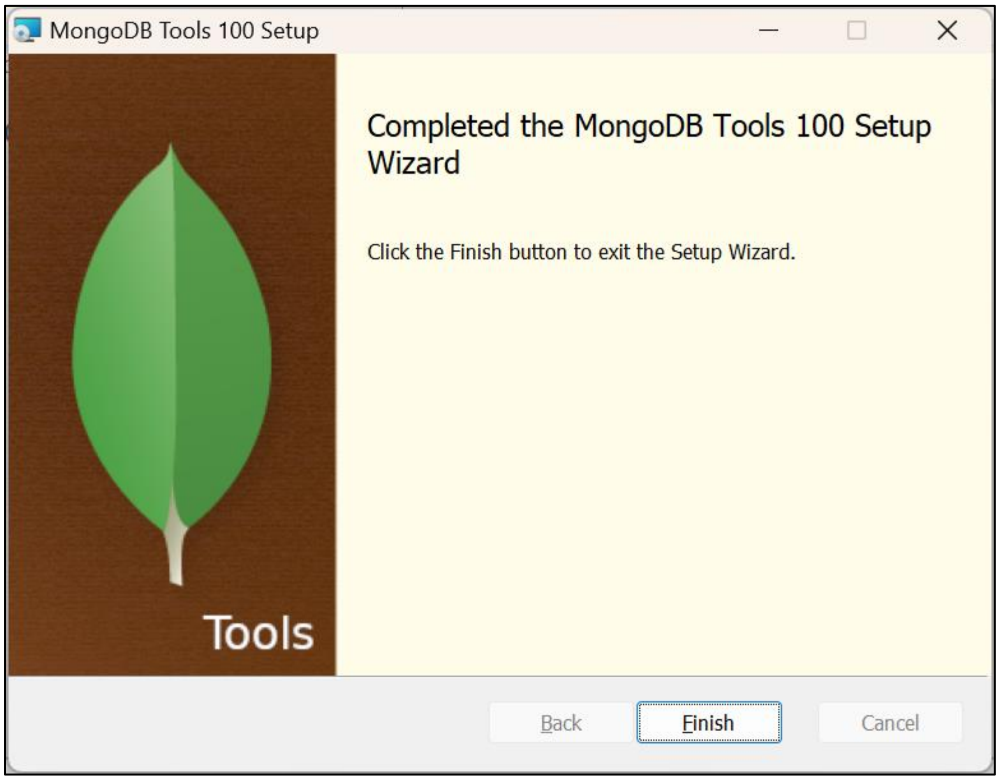
This feature requires 216MB on your hard drive.

Location: C:\Program Files\MongoDB\Tools\100\


MongoDB Tools 100 Setup

Ready to install MongoDB Tools 100

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.


MongoDB Tools 100 Setup

Completed the MongoDB Tools 100 Setup Wizard

Click the Finish button to exit the Setup Wizard.

Set the PATH variable

Identify MongoDB bin folder. By default, MongoDB installs to:

C:\Program Files\MongoDB\Server\8.2\bin

This is the folder containing the executables (mongod.exe, mongosh.exe, mongoimport.exe, etc.).

Open Environment Variables Settings

1. Press **Win + R**, type:
2. sysdm.cpl

   and hit Enter.
3. Go to the **Advanced** tab → click **Environment Variables**.

Edit the PATH variable

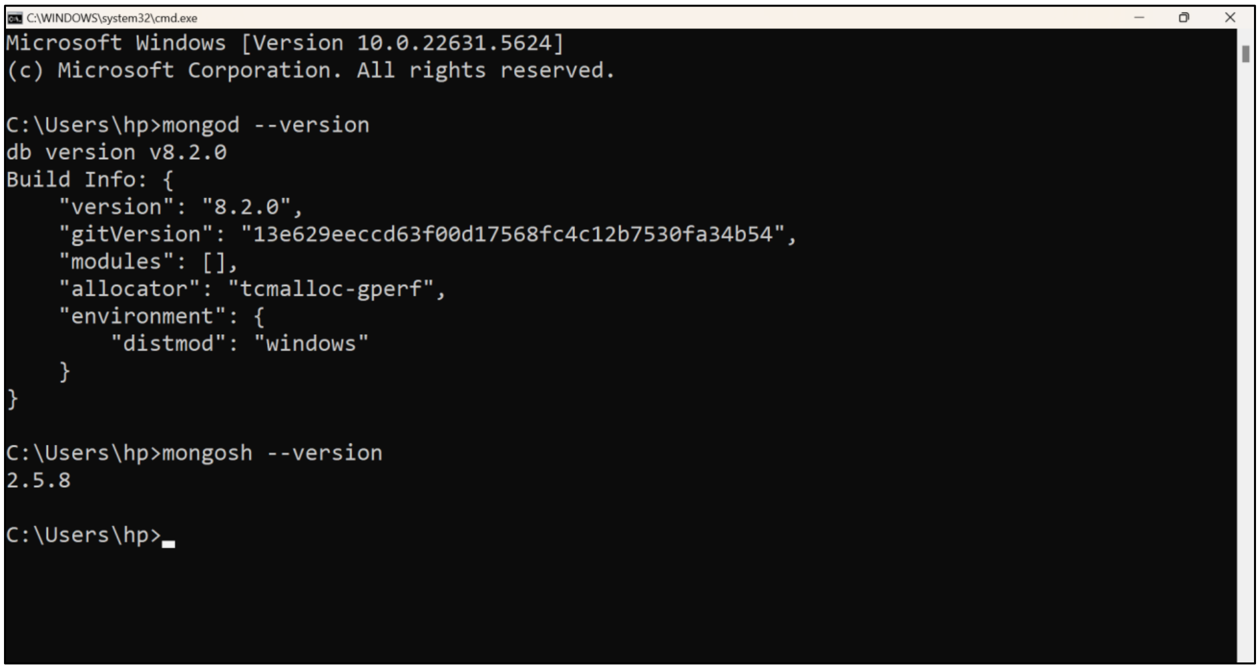
1. In the **System variables** section, find **Path** → select → click **Edit**.
2. Click **New** and add:
3. C:\Program Files\MongoDB\Server\8.2\bin
4. Click **OK** to save → close all windows.

Verify PATH is set

1. Open a **new Command Prompt** (important — changes apply to new sessions only).
2. Type:

   mongod --version

   mongosh --version
3. If it prints version info, the PATH is correctly set.



```
Microsoft Windows [Version 10.0.22631.5624]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>mongod --version
db version v8.2.0
Build Info: {
    "version": "8.2.0",
    "gitVersion": "13e629eeccd63f00d17568fc4c12b7530fa34b54",
    "modules": [],
    "allocator": "tcmalloc-gperf",
    "environment": {
        "distmod": "windows"
    }
}

C:\Users\hp>mongosh --version
2.5.8

C:\Users\hp>
```

Add Tools folder to PATH (recommended)

1. Press **Win** + **R** → type:

   sysdm.cpl

   → Enter.
2. Go to **Advanced** → **Environment Variables**.
3. Under **System variables**, find **Path** → Edit → **New** → paste:

   C:\Program Files\MongoDB\Tools\100\bin
4. Save and restart Command Prompt.
5. Test:

   mongoimport --version

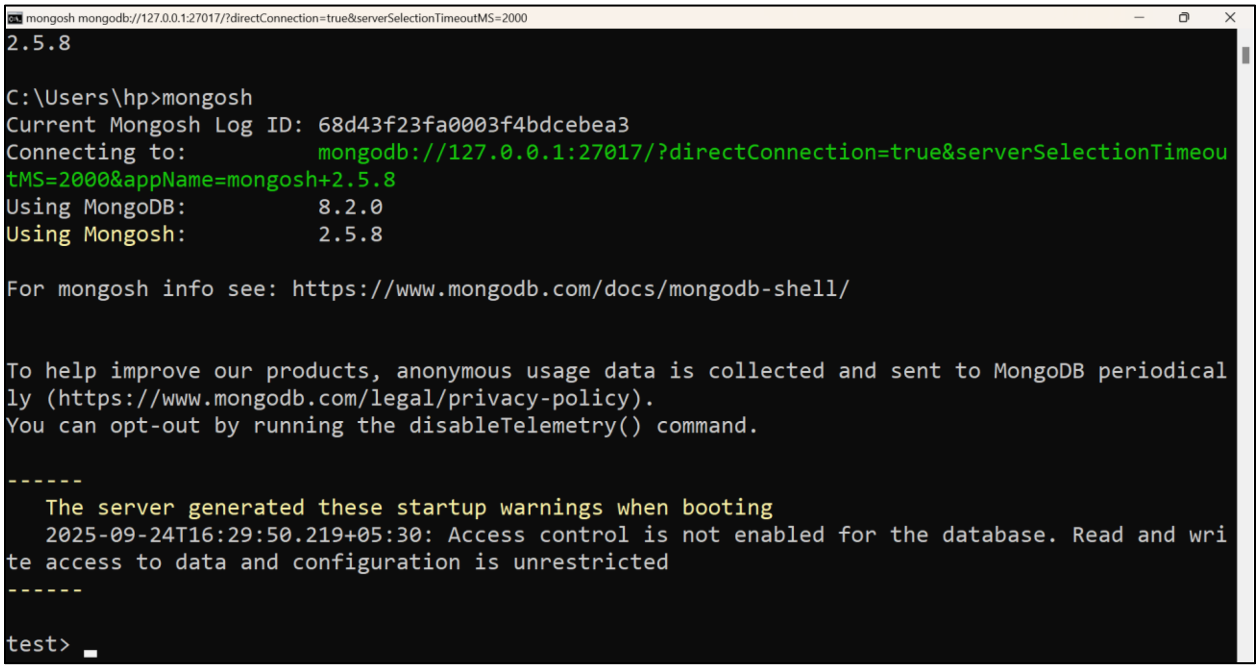
```
C:\Users\hp>mongoimport --version
mongoimport version: 100.13.0
git version: 23008ff975be028544710a5da6ae749dc7e90ab7
Go version: go1.23.8
   os: windows
   arch: amd64
   compiler: gc
```

Start the shell (connect locally)

Once mongod is running, open the Mongo shell:

> mongosh

By default mongosh connects to **mongodb://localhost:27017**.



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
2.5.8

C:\Users\hp>mongosh
Current Mongosh Log ID: 68d43f23fa0003f4bdcebea3
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeou
tMS=2000&appName=mongosh+2.5.8
Using MongoDB:          8.2.0
Using Mongosh:          2.5.8

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodical
ly (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2025-09-24T16:29:50.219+05:30: Access control is not enabled for the database. Read and wri
te access to data and configuration is unrestricted
------

test> _
```

Basic Data Operations (insert / find / query)

1. Create / switch DB and a collection

```
test> use db1
switched to db db1
```

2. Insert a single document

```
db1> db.users.insertOne({
     name: "Alice",
     age: 30,
     email: "alice@example.com",
     tags: ["admin","sales"]
})
{
  acknowledged: true,
  insertedId: ObjectId('68d43fa7fa0003f4bdcebea4')
}
```

3. Insert multiple documents

```
db1> db.users.insertMany([
       { name: "Bob", age: 25, email:"bob@example.com", tags: ["support"] },
       { name: "Carol", age: 35, email:"carol@example.com", tags: ["sales"] }
   ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68d44070fa0003f4bdcebea5'),
    '1': ObjectId('68d44070fa0003f4bdcebea6')
  }
}
```

4. Find documents

find() returns a cursor; mongosh prints the first batch automatically. For programmatic access use cursor methods.

Find all documents

```
db1> db.users.find()            // returns up to 20 docs in mongosh by default
```

```
[
  {
    _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
    name: 'Alice',
    age: 30,
    email: 'alice@example.com',
    tags: [ 'admin', 'sales' ]
  },
  {
    _id: ObjectId('68d44070fa0003f4bdcebea5'),
    name: 'Bob',
    age: 25,
    email: 'bob@example.com',
    tags: [ 'support' ]
  },
  {
    _id: ObjectId('68d44070fa0003f4bdcebea6'),
    name: 'Carol',
    age: 35,
    email: 'carol@example.com',
    tags: [ 'sales' ]
  }
]

db1> db.users.find().pretty()      // nicely formatted
[
  {
    _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
    name: 'Alice',
    age: 30,
    email: 'alice@example.com',
    tags: [ 'admin', 'sales' ]
  },
  {
    _id: ObjectId('68d44070fa0003f4bdcebea5'),
    name: 'Bob',
    age: 25,
    email: 'bob@example.com',
    tags: [ 'support' ]
  },
  {
    _id: ObjectId('68d44070fa0003f4bdcebea6'),
    name: 'Carol',
    age: 35,
    email: 'carol@example.com',
    tags: [ 'sales' ]
  }
]
```

Find single document

```
db1> db.users.findOne({ name: "Alice" })
{
  _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
  name: 'Alice',
  age: 30,
  email: 'alice@example.com',
  tags: [ 'admin', 'sales' ]
}
```

5. Filter / projection / sort / limit examples

Filter: age greater than 26

```
db1> db.users.find({ age: { $gt: 26 } })
[
  {
    _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
    name: 'Alice',
    age: 30,
    email: 'alice@example.com',
    tags: [ 'admin', 'sales' ]
  },
  {
    _id: ObjectId('68d44070fa0003f4bdcebea6'),
    name: 'Carol',
    age: 35,
    email: 'carol@example.com',
    tags: [ 'sales' ]
  }
]
```

Filter documents that have tag "admin"

```
db1> db.users.find({ tags: "admin" })
[
  {
    _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
    name: 'Alice',
    age: 30,
    email: 'alice@example.com',
    tags: [ 'admin', 'sales' ]
  }
]
```

Projection: only show email, hide _id

```
db1> db.users.find({ age: { $gt: 26 } }, { email: 1, _id: 0 })
[ { email: 'alice@example.com' }, { email: 'carol@example.com' } ]
```

Sort by age descending, limit 5

```
db1> db.users.find({}).sort({ age: -1 }).limit(5)
[
  {
    _id: ObjectId('68d44070fa0003f4bdcebea6'),
    name: 'Carol',
    age: 35,
    email: 'carol@example.com',
    tags: [ 'sales' ]
  },
  {
    _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
    name: 'Alice',
    age: 30,
    email: 'alice@example.com',
    tags: [ 'admin', 'sales' ]
  },
  {
    _id: ObjectId('68d44070fa0003f4bdcebea5'),
    name: 'Bob',
    age: 25,
    email: 'bob@example.com',
    tags: [ 'support' ]
  }
]
```

6. Update documents

Update a single doc

```
db1> db.users.updateOne({ name: "Alice" }, { $set: { age: 31 } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}

db1> db.users.findOne({ name: "Alice" })
{
  _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
  name: 'Alice',
  age: 31,
  email: 'alice@example.com',
  tags: [ 'admin', 'sales' ]
}
```

Update many documents

```
db1> db.users.updateMany({ tags: "sales" }, { $set: { status: "active" } })
{
 acknowledged: true,
 insertedId: null,
 matchedCount: 2,
 modifiedCount: 2,
 upsertedCount: 0
}
db1> db.users.find()
[
  {
   _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
   name: 'Alice',
   age: 31,
   email: 'alice@example.com',
   tags: [ 'admin', 'sales' ],
   status: 'active'
  },
  {
   _id: ObjectId('68d44070fa0003f4bdcebea5'),
   name: 'Bob',
   age: 25,
   email: 'bob@example.com',
   tags: [ 'support' ]
  },
  {
   _id: ObjectId('68d44070fa0003f4bdcebea6'),
   name: 'Carol',
   age: 35,
   email: 'carol@example.com',
   tags: [ 'sales' ],
   status: 'active'
  }
]
```

7. Delete documents

deleteOne() removes first match;

deleteMany() removes all matches. Always be careful with empty filters ({}) — that will delete everything.

```
db1> db.users.deleteOne({ name: "Bob" })
{ acknowledged: true, deletedCount: 1 }

db1> db.users.deleteMany({ status: "inactive" })
{ acknowledged: true, deletedCount: 0 }

db1> db.users.find()
[
  {
    _id: ObjectId('68d43fa7fa0003f4bdcebea4'),
    name: 'Alice',
    age: 31,
    email: 'alice@example.com',
    tags: [ 'admin', 'sales' ],
    status: 'active'
  },
  {
    _id: ObjectId('68d44070fa0003f4bdcebea6'),
    name: 'Carol',
    age: 35,
    email: 'carol@example.com',
    tags: [ 'sales' ],
    status: 'active'
  }
]
```

8. Importing data using mongoimport

mongoimport is a command-line tool (part of Database Tools) — run it from shell (not inside mongosh).

Example 1: import JSON array

Create **users.json** document in C:\data\ folder

```
[
  { "name": "Bob", "age": 25, "email": "bob@example.com" },
  { "name": "Alice", "age": 30, "email": "alice@example.com" },
  { "name": "John", "age": 28, "email": "john@example.com" }
]
```

Import command:

```
C:\Users\hp> "C:\Program Files\MongoDB\Tools\100\bin\mongoimport.exe" --db db1 --collection users --file "C:\data\users.json" --jsonArray
2025-09-25T01:52:08.102+0530    connected to: mongodb://localhost/
2025-09-25T01:52:08.106+0530    3 document(s) imported successfully. 0 document(s) failed to import.
```

Verify after import

In mongosh:

```
test> use db1
switched to db db1

db1> db.users.countDocuments()
3

db1> db.users.find().limit(5).pretty()
[
  {
    _id: ObjectId('68d452f08fe17e7b99bfe2ed'),
    name: 'John',
    age: 28,
    email: 'john@example.com'
  },
  {
    _id: ObjectId('68d452f08fe17e7b99bfe2ee'),
    name: 'Alice',
    age: 30,
    email: 'alice@example.com'
  },
  {
    _id: ObjectId('68d452f08fe17e7b99bfe2ef'),
    name: 'Bob',
    age: 25,
    email: 'bob@example.com'
  }
]
```

Example 2: import CSV

Create **users1.csv** document in C:\data\ folder.

```
name,age,email
Alice,30,alice@example.com
Bob,25,bob@example.com
```

Import command:

```
C:\Users\hp>"C:\Program Files\MongoDB\Tools\100\bin\mongoimport.exe" --db db1 --collection users1 --type csv --headerline --file "C:\data\users1.csv"
2025-09-25T02:05:15.504+0530    connected to: mongodb://localhost/
2025-09-25T02:05:15.524+0530    2 document(s) imported successfully. 0 document(s) failed to import.
```

Verify after import

In mongosh:

```
test> use db1
switched to db db1
db1> db.users1.countDocuments()
2

db1> db.users1.find().limit(5).pretty()
[
  {
    _id: ObjectId('68d4560352db82ac983cd61f'),
    name: 'Alice',
    age: 30,
    email: 'alice@example.com'
```

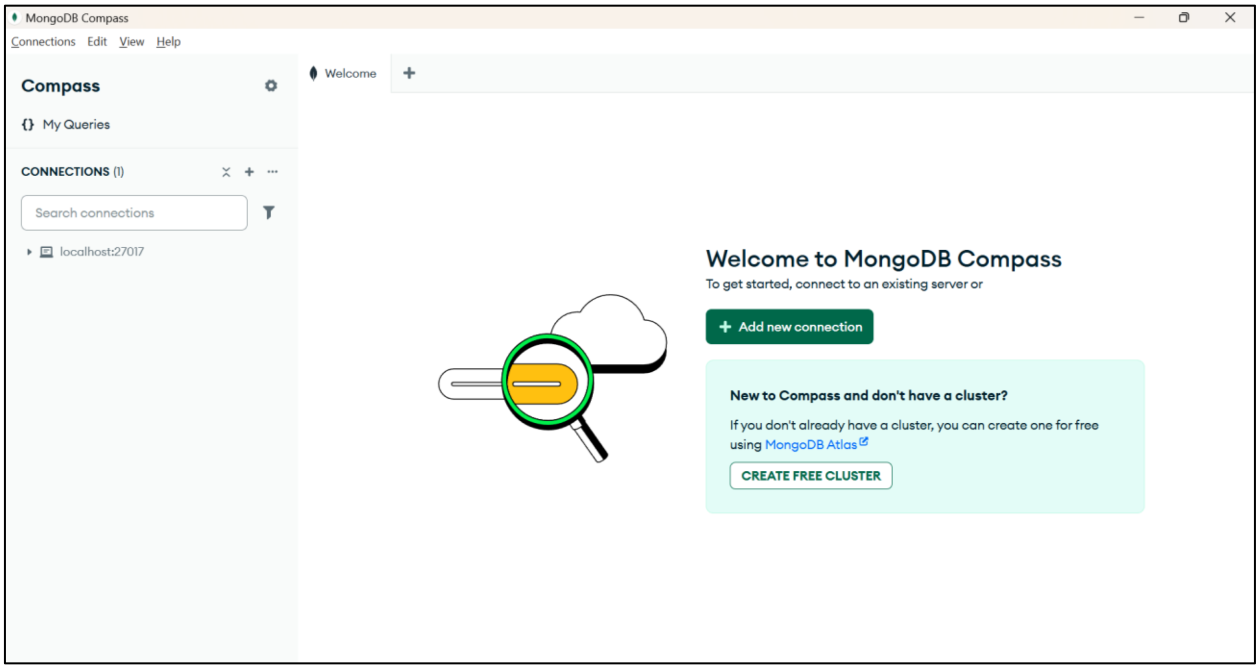
```
 },
 {
  _id: ObjectId('68d4560352db82ac983cd620'),
  name: 'Bob',
  age: 25,
  email: 'bob@example.com'
 }
]
```

Import via GUI (MongoDB Compass)

In **MongoDB Compass,** we can import JSON/CSV files via its Import dialog.

Open MongoDB Compass

1. Launch **MongoDB Compass**.



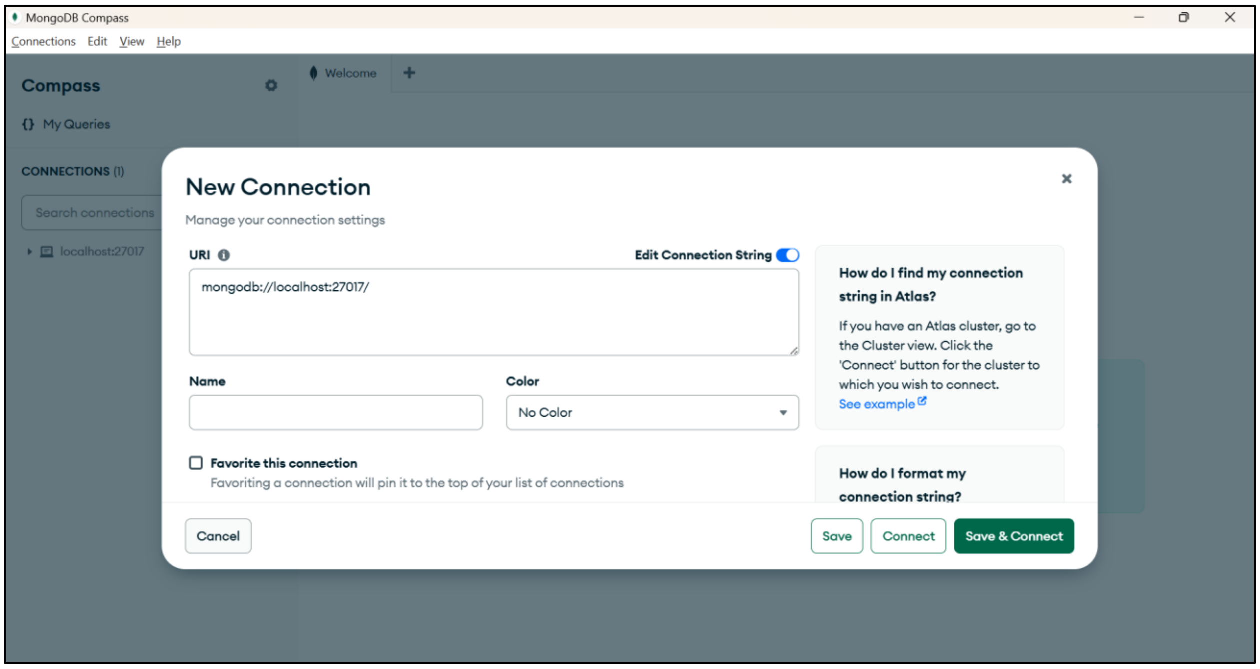
Welcome to MongoDB Compass

To get started, connect to an existing server or

New to Compass and don't have a cluster?

If you don't already have a cluster, you can create one for free using MongoDB Atlas

2. Connect to your MongoDB server:
   - **Hostname:** localhost
   - **Port:** 27017
   - Click **Connect**.



New Connection

Manage your connection settings

URI — Edit Connection String

mongodb://localhost:27017/

Name

Color: No Color

Favorite this connection

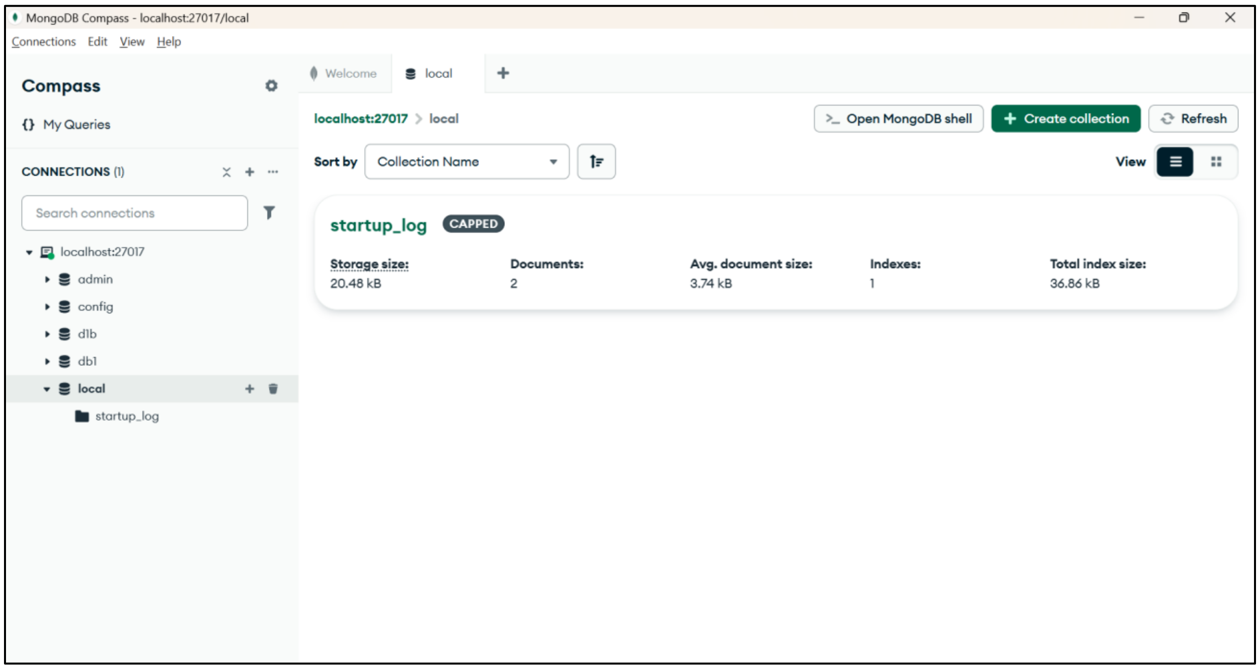
Favoriting a connection will pin it to the top of your list of connections

How do I find my connection string in Atlas?

If you have an Atlas cluster, go to the Cluster view. Click the 'Connect' button for the cluster to which you wish to connect.

See example

How do I format my connection string?



localhost:27017 > local

Sort by: Collection Name

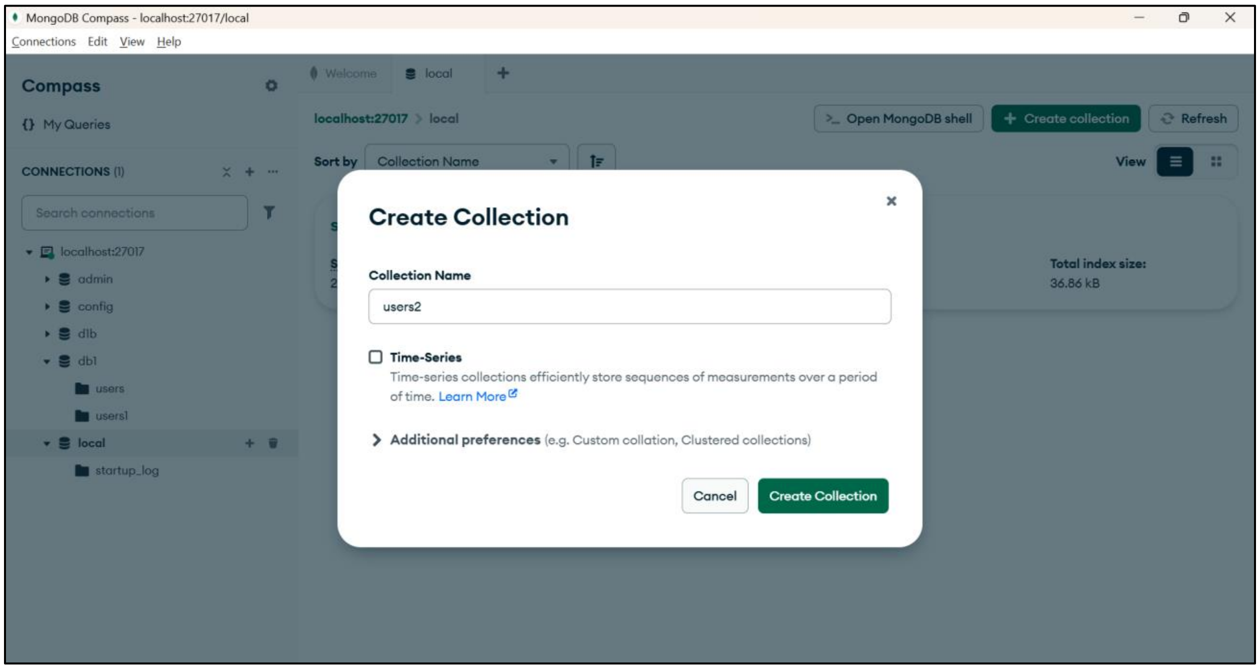
startup_log CAPPED

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
20.48 kB	2	3.74 kB	1	36.86 kB

Select or Create Database

1. In Compass, you'll see the list of databases.
2. To use an existing database, click it (e.g., db1).
3. To create a new database:
    - Click **"Create Database"**.
    - Enter:

        ▪ **Database Name:** db1
        ▪ **Collection Name:** users
    ○ Click **Create Database**.



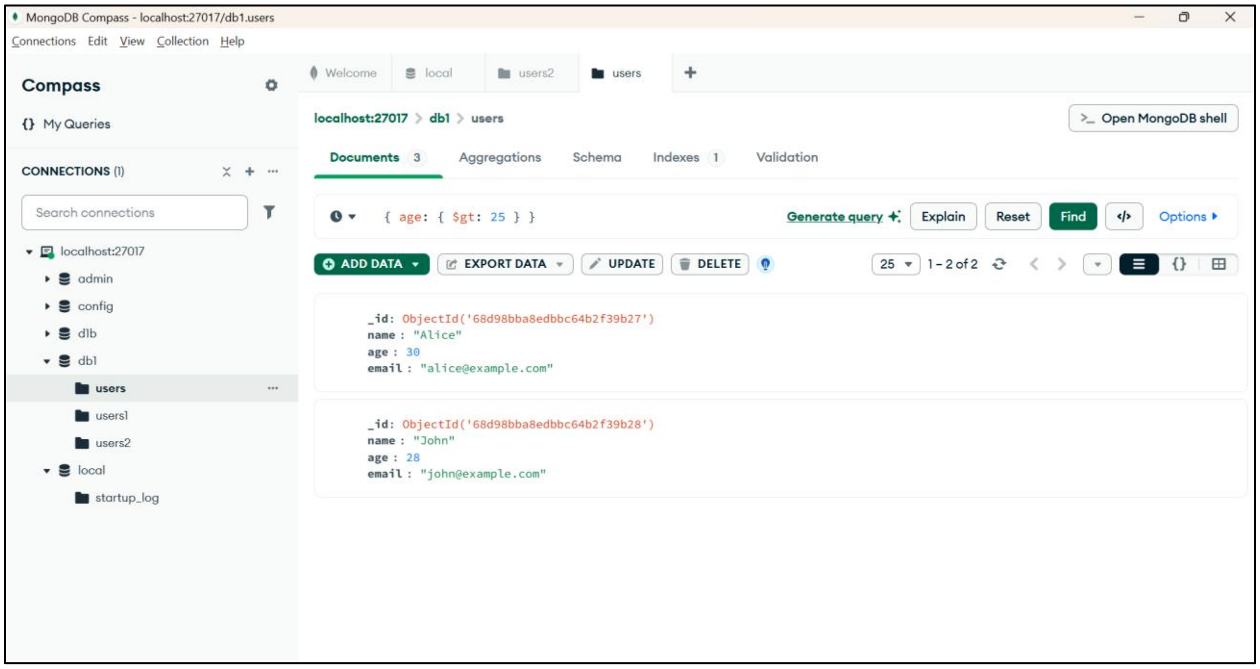
Import Data into a Collection

1. Click the **collection name** (e.g., users) in the left panel.
2. Click **"Add Data"** → **"Import File"**.
3. In the dialog:
    ○ **Select File:** Browse and choose your file (users.json or users.csv).
    ○ **File Type:** JSON or CSV (choose based on your file).
    ○ **JSON Array:** Check this if your JSON file has [...] at the top level.
    ○ **CSV Options:** If CSV, you can choose **Header Line** or map columns manually.
4. Click **Import**.

Verify Imported Data

After import:

1. The collection will display all documents in a table view.
2. You can click **"Documents"** tab to see each document.
3. You can also run queries in the **Filter** bar:
4. { age: { $gt: 25 } }
   → shows all users with age > 25.



```
{ age: { $gt: 25 } }
```

```
_id: ObjectId('68d98bba8edbbc64b2f39b27')
name : "Alice"
age : 30
email : "alice@example.com"
```

```
_id: ObjectId('68d98bba8edbbc64b2f39b28')
name : "John"
age : 28
email : "john@example.com"
```

Example Scenario

- File: users.json

```
[
  { "name": "Bob", "age": 25, "email": "bob@example.com" },
  { "name": "Alice", "age": 30, "email": "alice@example.com" },
  { "name": "John", "age": 28, "email": "john@example.com" }
]
```

- Import into **db1.users** using JSON Array option checked.
- After import, in Compass, **Documents tab** shows all 3 users.