

# 2

## Installing MongoDB

# Objectives

After completing this lesson, you should be able to do the following:

- Installation of MongoDB and MongoDB Compass
- Working with MongoShell
- Configuration
- Collections and Documents
- Data Types





# Installing MongoDB

- The installers for MongoDB are available in both the 32-bit and 64-bit format.
- The 32-bit installers are good for development and test environments.
- But for production environments you should use the 64-bit installers. Otherwise, you can be limited to the amount of data that can be stored within MongoDB.
- It is advisable to always use the stable release for production environments.

# Installation : Step 1

- Download MongoDB Community Server
  - [https://www.mongodb.com/try/download/community]

The screenshot shows the MongoDB Community Server download page. At the top, there's a navigation bar with links for Products, Solutions, Resources, Company, Pricing, Sign In, and Try Free. Below the navigation, there's a section titled "MongoDB Community Server" with a brief description of its features. To the right, there's a sidebar titled "Available Downloads" with dropdown menus for Version (set to 5.0.2), Platform (set to Windows), and Package (set to msi). A large green "Download" button is at the bottom of this sidebar. Red arrows from the list above point to the "Download" button and the "Version" dropdown.

MongoDB Community Server

The Community version of our distributed database offers a flexible document data model along with support for ad-hoc queries, secondary indexing, and real-time aggregations to provide powerful ways to access and analyze your data.

The database is also offered as a fully-managed service with MongoDB Atlas. Get access to advanced functionality such as auto-scaling, serverless instances (in preview), full-text search, and data distribution across regions and clouds. Deploy in minutes on AWS, Google Cloud, and/or Azure, with no downloads necessary.

Give it a try with a free, highly-available 512 MB cluster.

Available Downloads

Version: 5.0.2 (current)

Platform: Windows

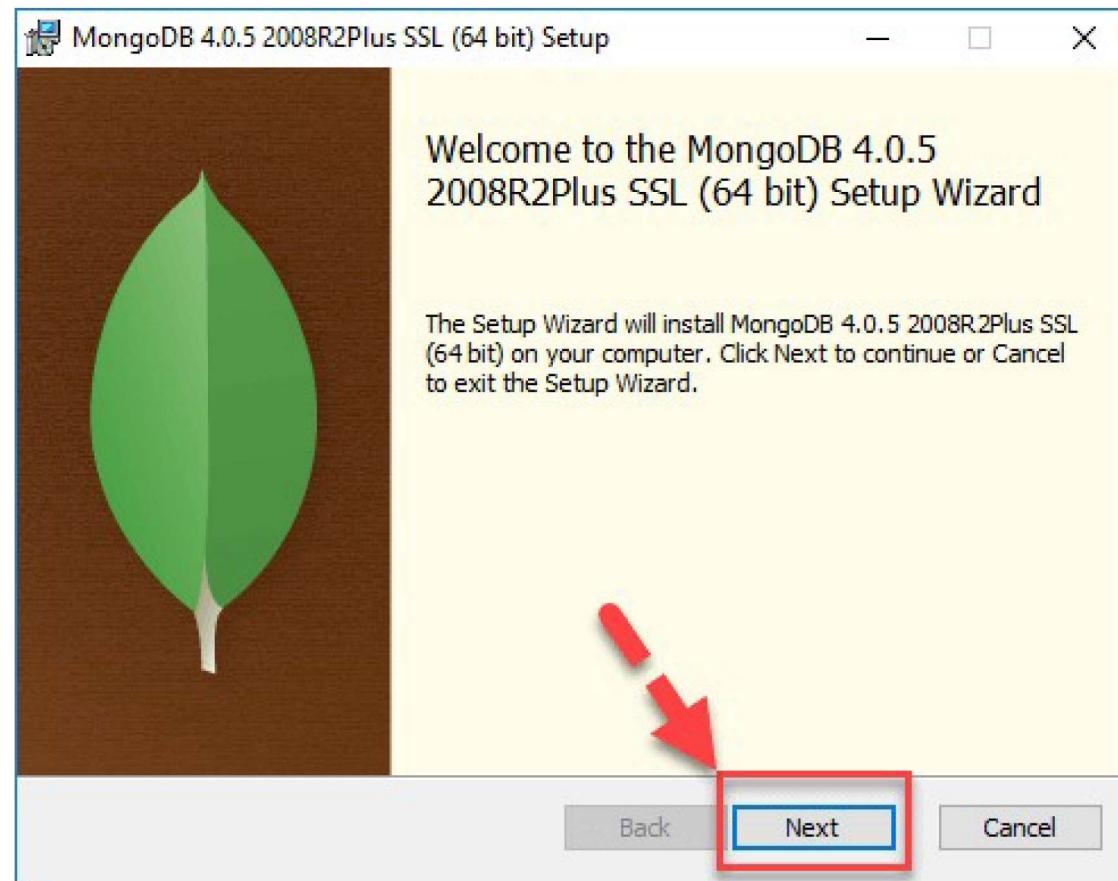
Package: msi

Download Copy Link

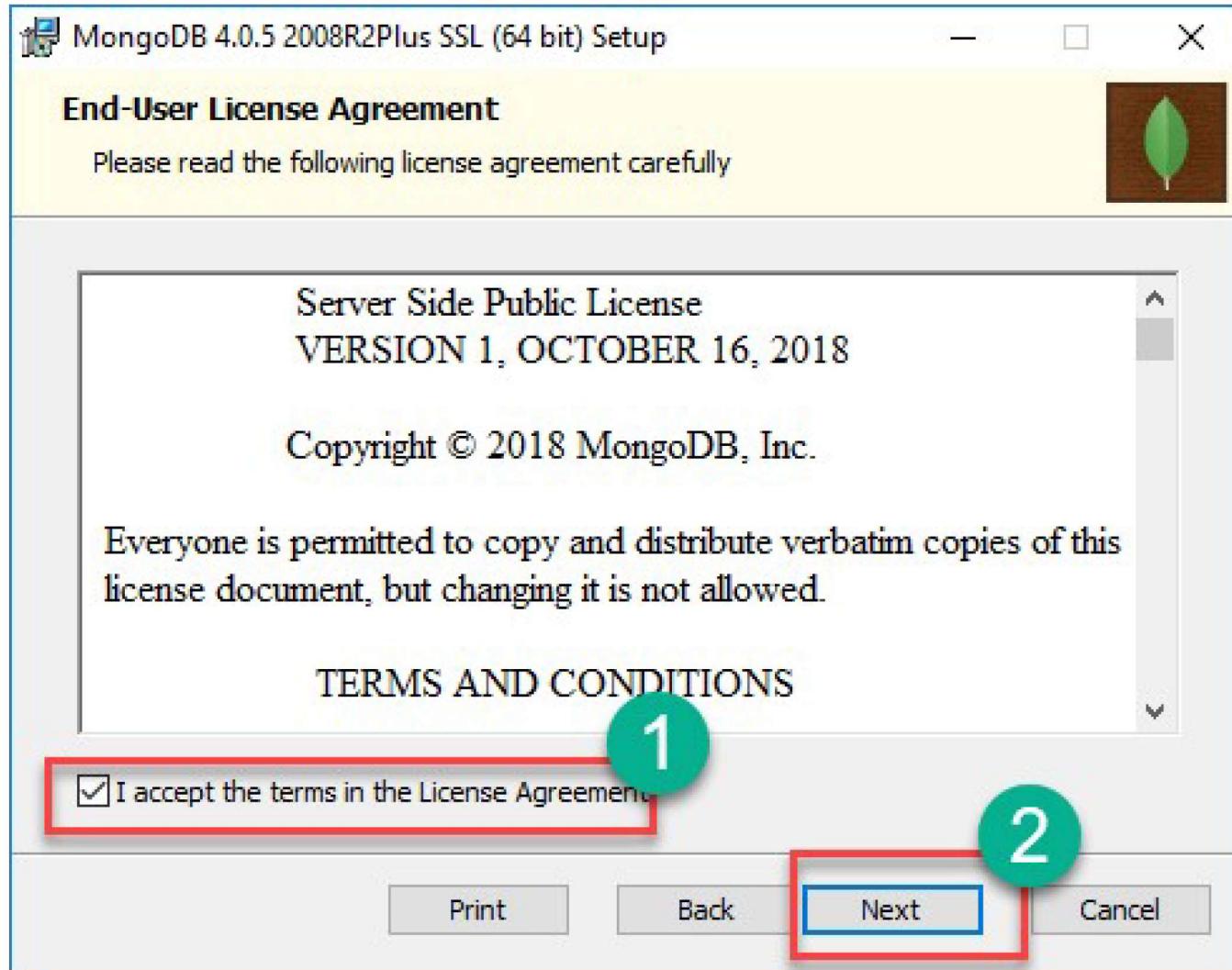
Current releases & packages

## Step 2 : Click on Setup

- Once download is complete open the msi file. Click Next in the start up screen

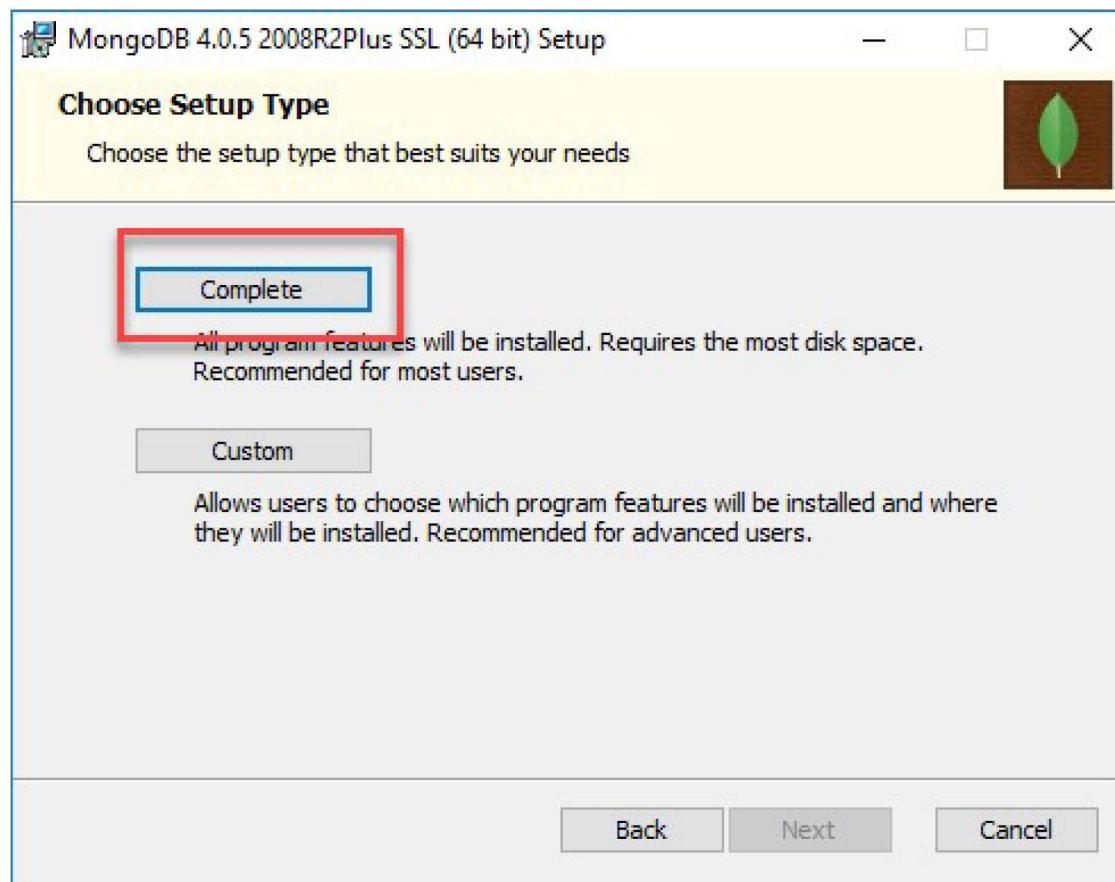


## Step 3 : Accept the End-User License Agreement



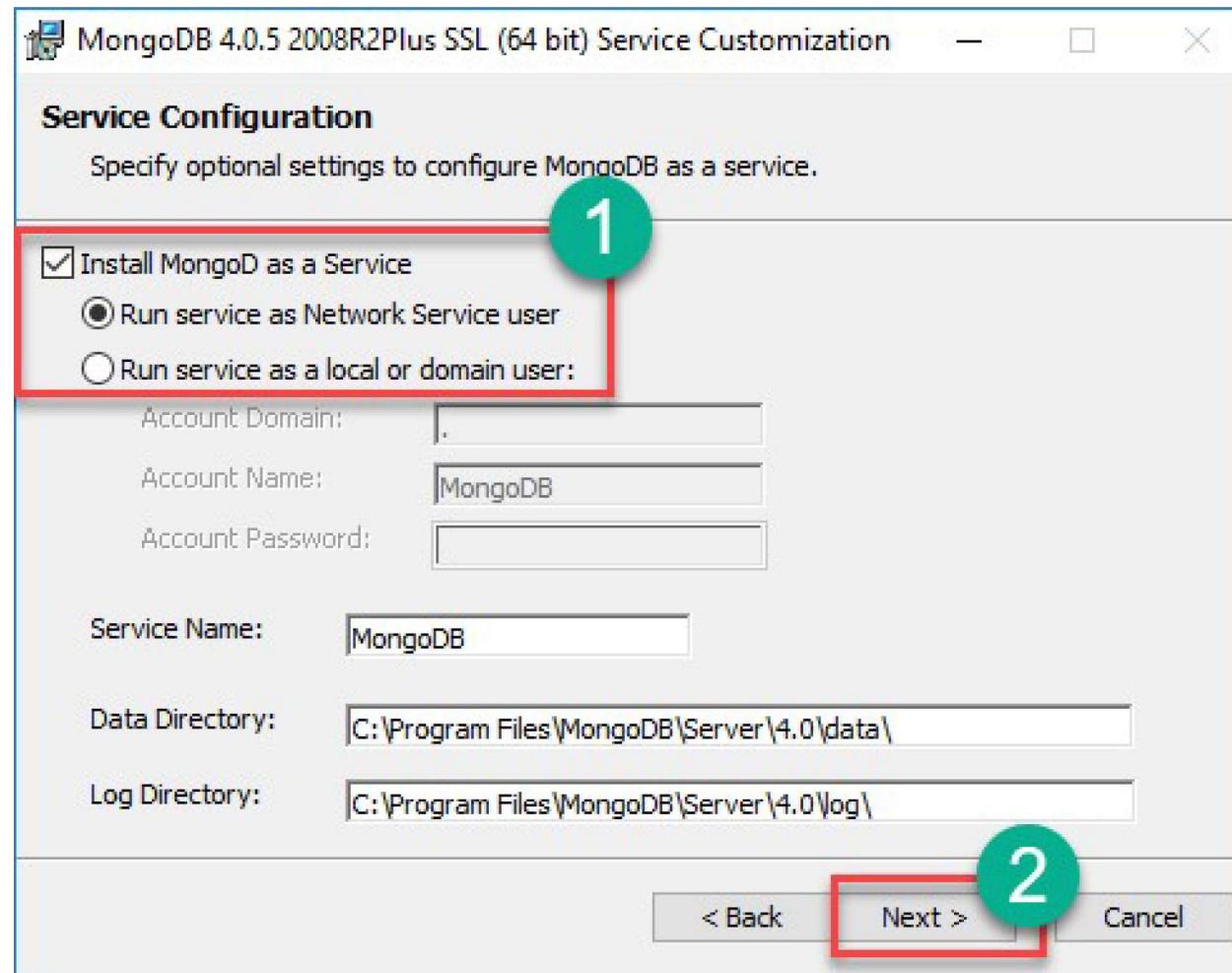
## Step 4 : Click on the “complete” button

- Click on the “complete” button to install all of the components. The custom option can be used to install selective components or if you want to change the location of the installation.

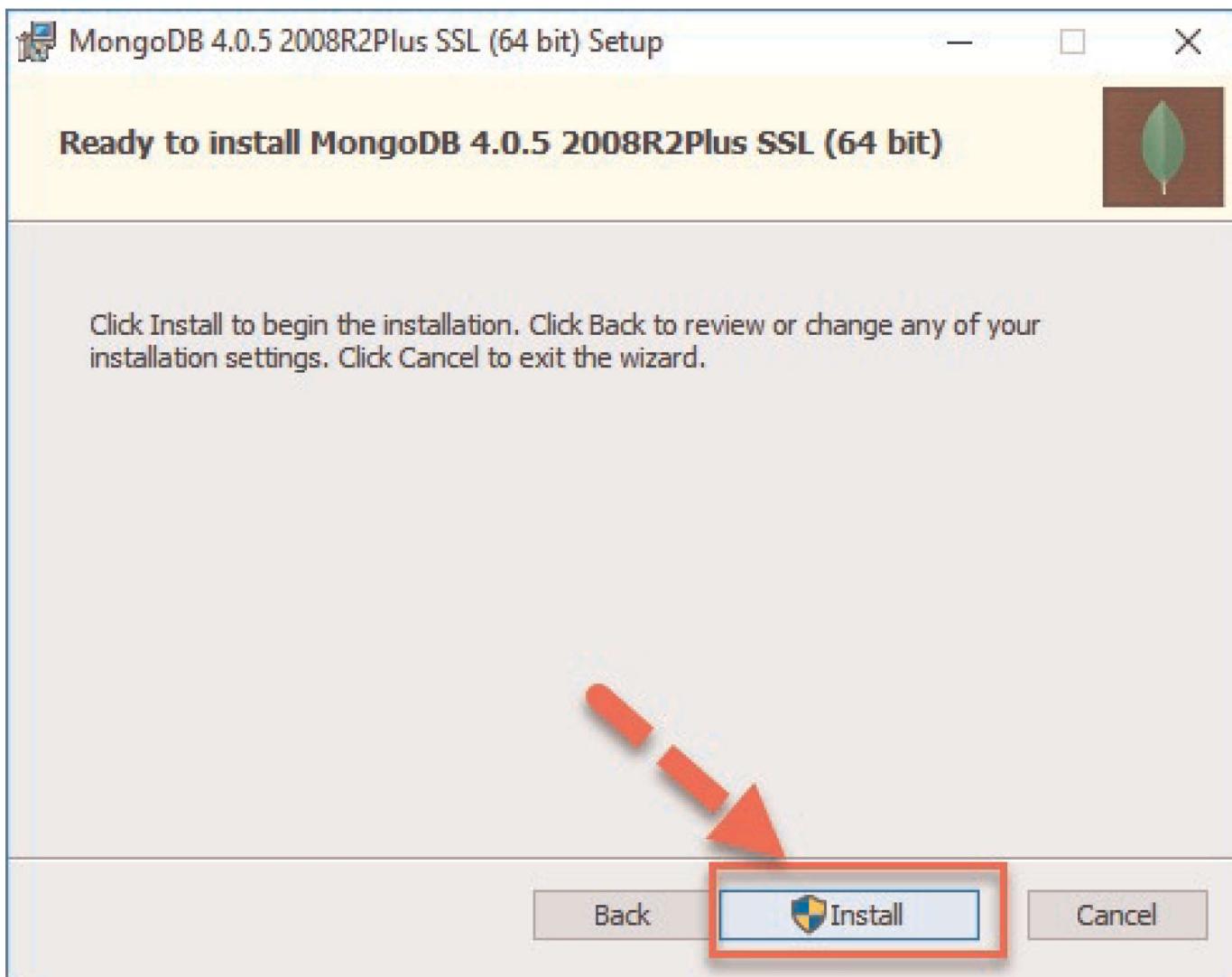


## Step 5 : Service Configuration

1. Select “Run service as Network Service user”. make a note of the data directory, we’ll need this later.
2. Click Next

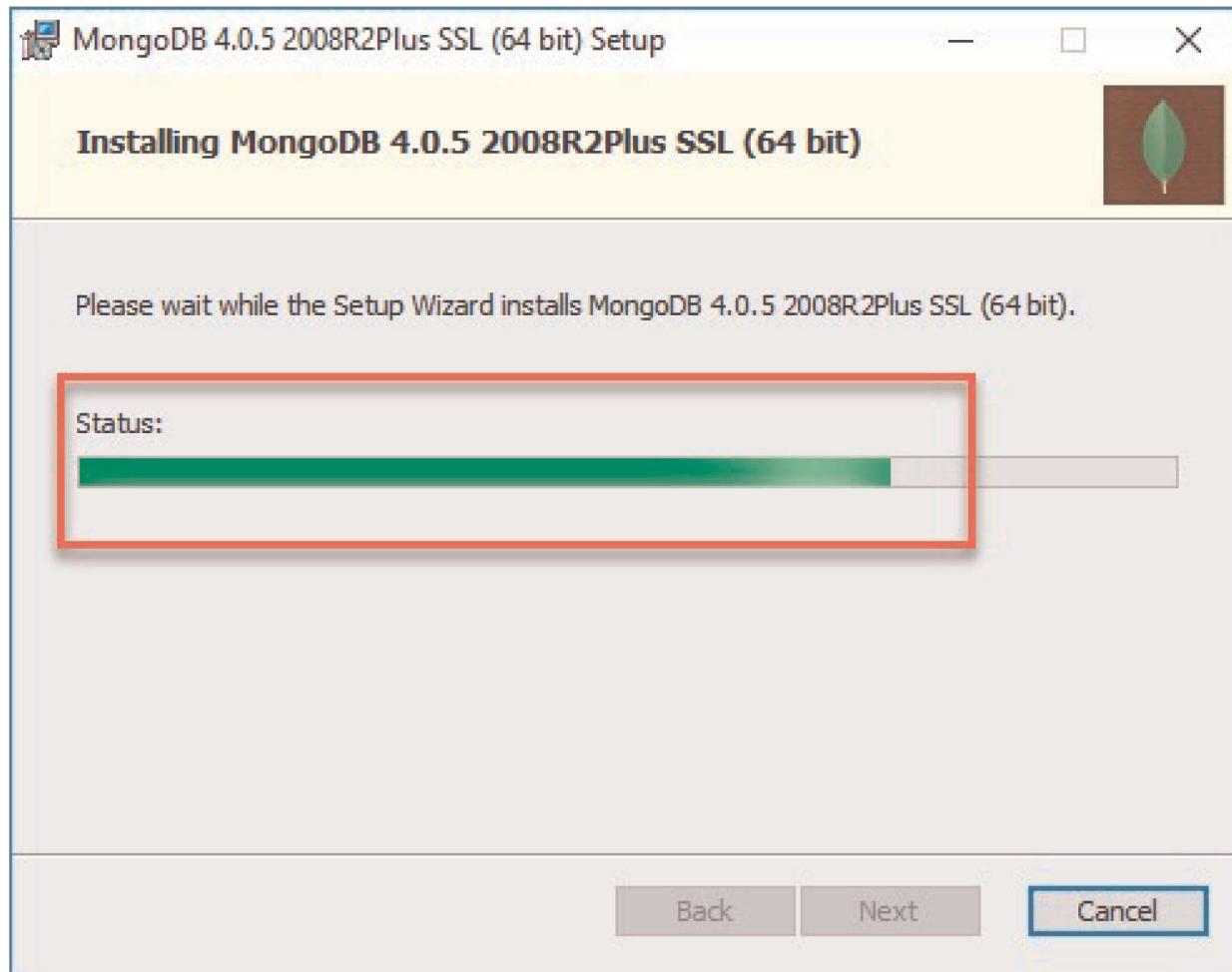


## Step 6 : Start installation process

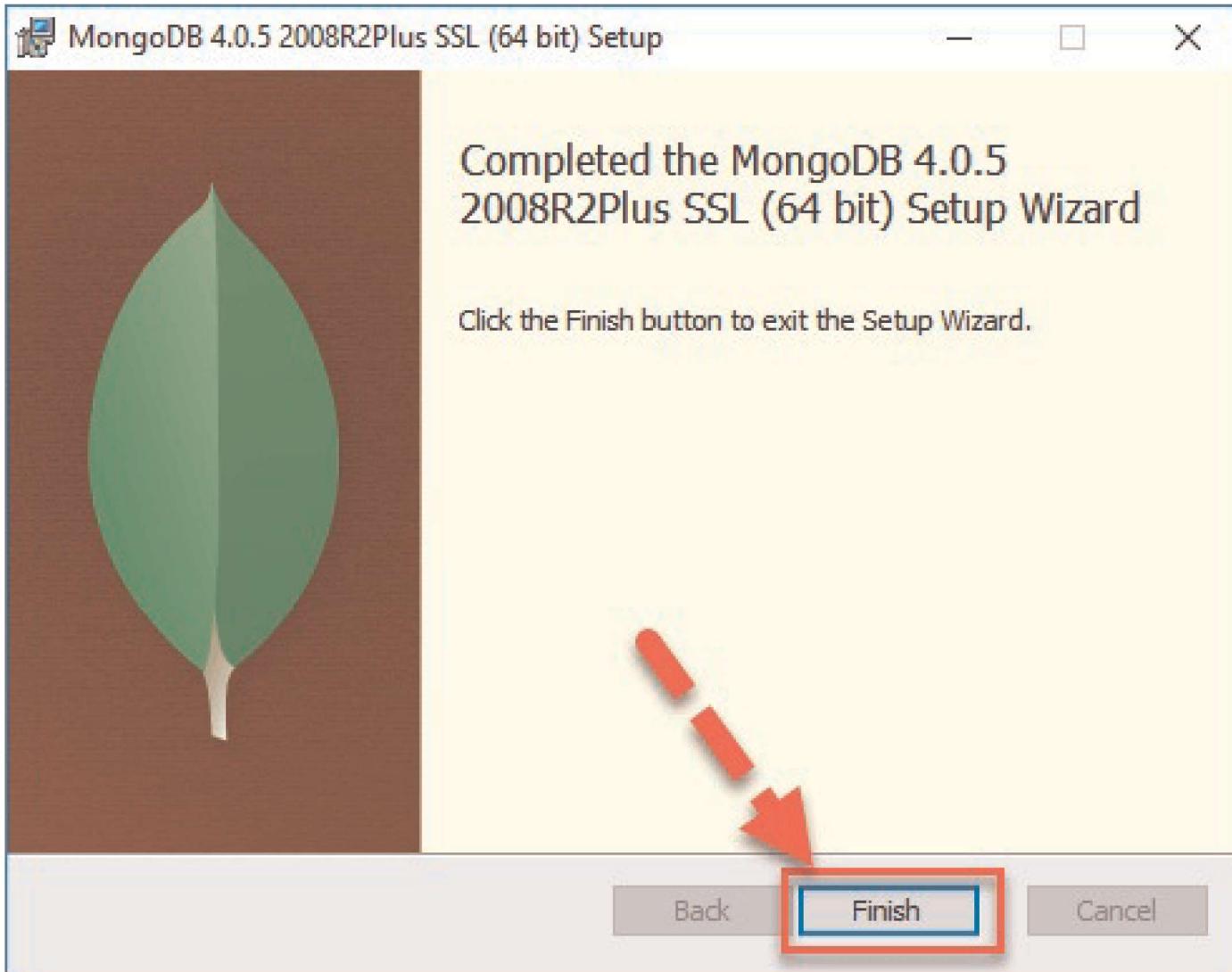


## Step 7: Click Next once completed

- Installation begins. Click Next once completed



## Step 8: Click on the Finish button





# Installing & Using MongoDB Compass

# Step 1 : MongoDB Compass gets Installed with installation of MongoDB Installer

The screenshot shows the MongoDB Compass application window. The title bar reads "MongoDB Compass - Connect". The menu bar includes "Connect", "View", and "Help". On the left, there's a sidebar with "New Connection", "Favorites", and "Recent". The main area is titled "Welcome to MongoDB Compass". It features several performance charts:

- OPERATIONS**: A line chart showing operations over time. Legend: INSERTS (blue), UPDATES (purple), DELETED (red), SELECTED (orange), COMMANDS (yellow), and DATABASES (green). Values: INSERTS (48), UPDATES (107), DELETED (89), SELECTED (93), COMMANDS (0), and DATABASES (0).
- READ & WRITE**: A line chart showing active reads, active writes, and queued reads/writes over time.
- NETWORK**: A line chart showing fast read, fast cost, and connections over time. Legend: FAST READ (blue), FAST COST (orange), and CONNECTIONS (purple). Values: FAST READ (48), FAST COST (90), and CONNECTIONS (16).
- MEMORY**: A line chart showing usage, memory used, and memory free over time. Legend: USED (blue), MEMORY USED (orange), and MEMORY FREE (green). Values: USED (3.77), MEMORY USED (0.82), and MEMORY FREE (0).

Annotations explain the charts:

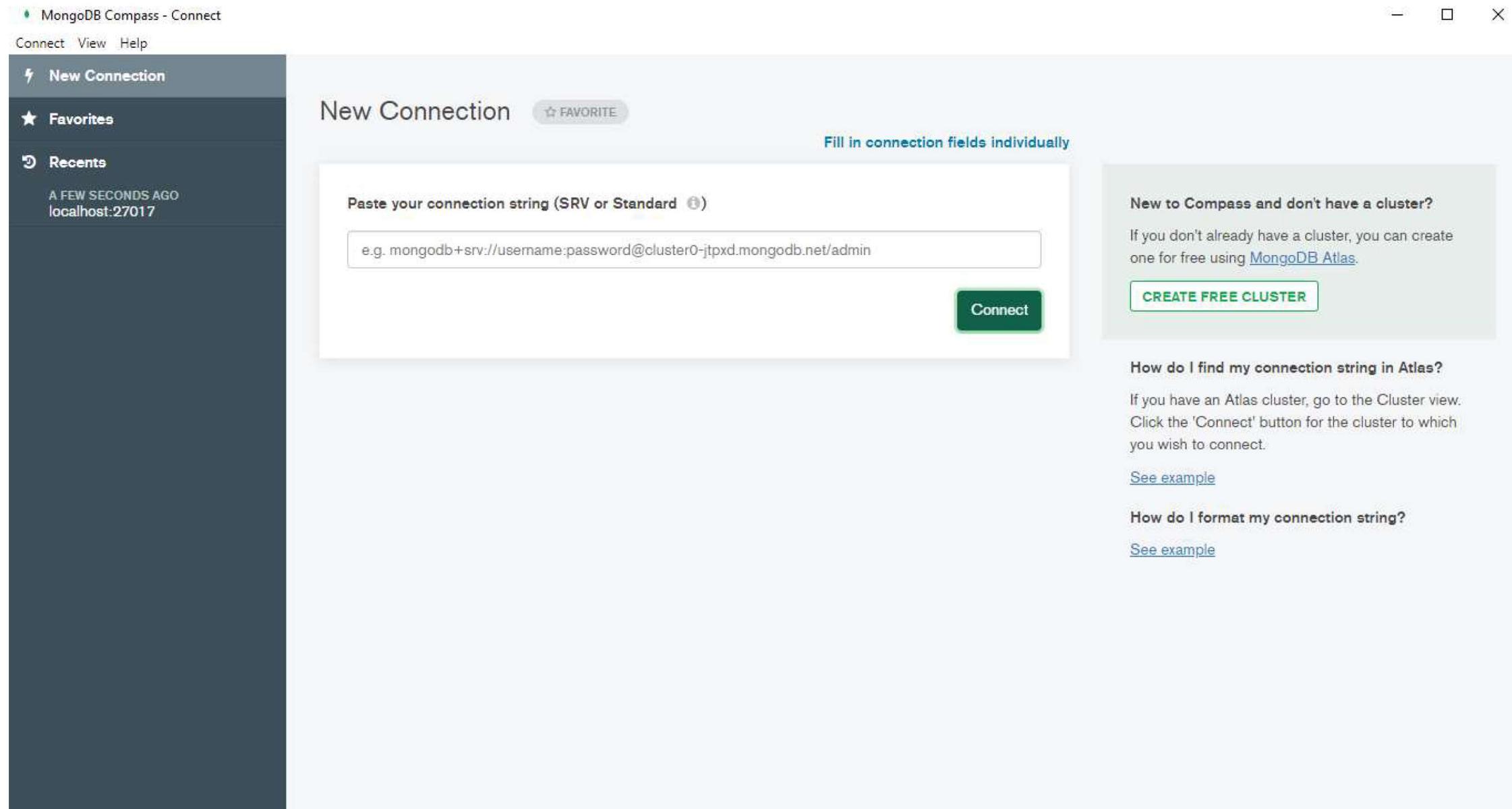
- The first annotation points to the top of the "OPERATIONS" chart with the text: "The charts show database operations, queues, network traffic, connections and memory over time."
- The second annotation points to the top of the "HOTTEST COLLECTIONS" section with the text: "The percentage value represents the system load for each collection."
- The third annotation points to the bottom of the "SLOWEST OPERATIONS" section with the text: "Click on a slow operation to examine more details."

On the right side of the screen, there's a "Performance Charts" section with the following text:

Real-time server statistics let you view key server metrics and database operations. Drill down into database operations easily and understand your most active collections.

At the bottom right, there are "Next >" and "?" buttons.

# Click : Connect



You will see homescreen with list of current databases.

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with a dark background and white text. It says "My Cluster" at the top, followed by "3 DBS" and "1 COLLECTIONS". Below that are buttons for "filter" and three database names: "admin", "config", and "local". A red callout bubble points from the text "List of current Databases" to the sidebar area. On the right, the main window has a light gray background. At the top, it shows "localhost:27017 STANDALONE". Below that is a navigation bar with tabs: "Databases" (which is selected and highlighted in green), "Performance", and "Sharding". There's also a "CREATE DATABASE" button. A red callout bubble points from the text "Click to create new DB" to this button. The main area displays a table of databases:

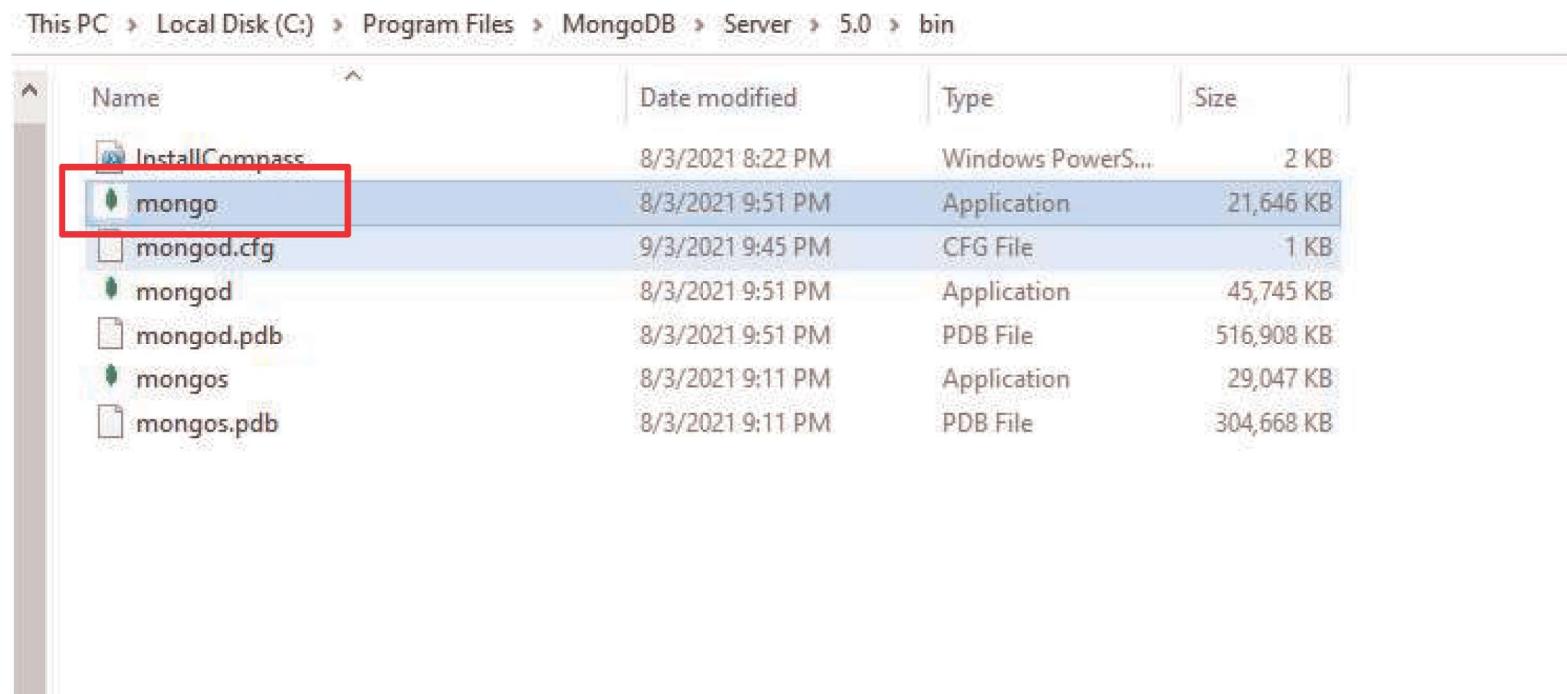
Database Name	Storage Size
admin	16.0KB
config	16.0KB
local	16.0KB

## Hello World MongoDB: JavaScript Driver

- Drivers in MongoDB are used for connectivity between client applications and the database. For example, if you had Java program and required it to connect to MongoDB then you would require to download and integrate the Java driver so that the program can work with the MongoDB database.
- The driver for JavaScript comes out of the box. The MongoDB shell which is used to work with MongoDB database is actually a javascript shell. To access it

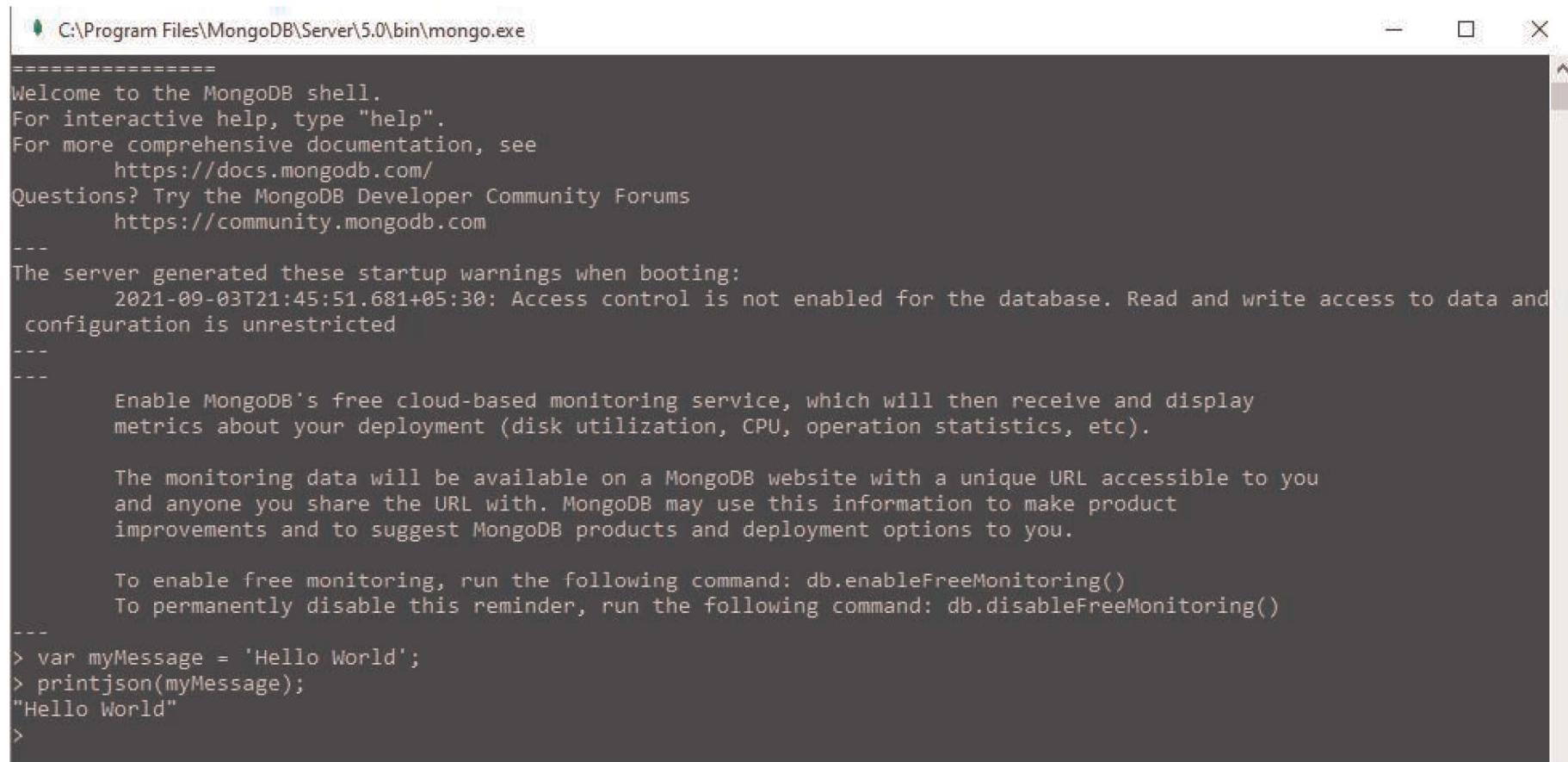
## Step 1 :

- Go to " C:\Program Files\MongoDB\Server\5.0\bin" and double click on mongo.exe. Alternatively, you can also click on the MongoDB desktop item



## Step 2 : Enter following program into shell

```
var myMessage='Hello World';
printjson(myMessage);
```



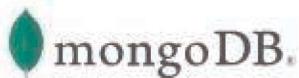
The screenshot shows a terminal window titled "C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe". The window displays the MongoDB shell startup message, which includes welcome text, help instructions, documentation links, and community forums. It also shows startup warnings about access control and monitoring. At the bottom of the window, the user has entered the provided JavaScript code into the shell, resulting in the output "Hello World".

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
=====
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
    https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
    https://community.mongodb.com
---
The server generated these startup warnings when booting:
    2021-09-03T21:45:51.681+05:30: Access control is not enabled for the database. Read and write access to data and
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 you
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()
---
> var myMessage = 'Hello World';
> printjson(myMessage);
"Hello World"
>
```

# MongoDB Database Tools



Products

Solutions

Resources

Company

Pricing



Sign In

Try Free

## MongoDB Database Tools

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.

### Available Downloads

Version

100.5.0

Platform

Windows x86\_64

Package

zip

Download

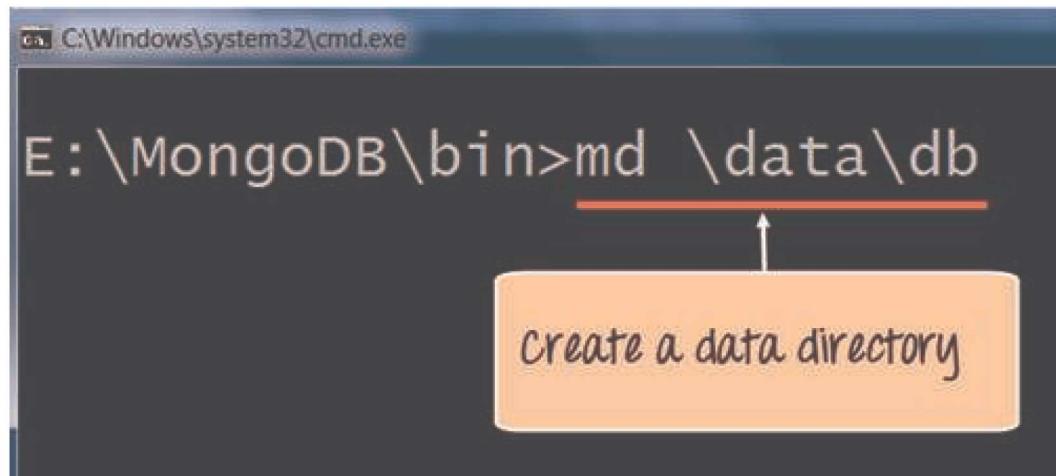
Copy Link

Documentation

Archived releases

# MongoDB Configuration, Import, and Export

- Before starting the MongoDB server, the first key aspect is to configure the data directory where all the MongoDB data will be stored.



```
C:\Windows\system32\cmd.exe
E:\MongoDB\bin>md \data\db
↑
create a data directory
```

- The above command ‘`md \data\db`’ makes a directory called `\data\db` in your current location.
- MongoDB will automatically create the databases in this location, because this is the default location for MongoDB to store its information. We are just ensuring the directory is present, so that MongoDB can find it when it starts.

The import of data into MongoDB is done using the “mongoimport” command.

**Step 1:** Create a CSV file called data.csv and put the following data in it

Employeeid,EmployeeName

1,Rahul

2,Imran

3,Smith

we are assuming we want to import 3 documents into a collection called data. The first row is called the header line which will become the Field names of the collection.

```
E:\MongoDB\bin>mongoimport --db TestDB --type csv --headerline --file data.csv
```

specify the database the  
data needs to imported in

specify that we are  
importing an csv file

Name of the csv file

The screenshot shows a Windows desktop environment. At the top, there is a taskbar with several pinned icons. Below the taskbar, two windows are open:

- Notepad++**: The title bar says "F:\MongoDB Tools\bin\data.csv - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and ?.
- Command Prompt**: The title bar says "C:\WINDOWS\system32\cmd.exe". The window contains the following text output from the mongoimport command:

```
F:\MongoDB Tools\bin>mongoimport --db TestDB --type csv --headerline --file data.csv
2021-09-03T22:35:43.699+0530      no collection specified
2021-09-03T22:35:43.700+0530      using filename 'data' as collection
2021-09-03T22:35:44.325+0530      connected to: mongodb://localhost/
2021-09-03T22:35:44.377+0530      3 document(s) imported successfully. 0 document(s) failed to import.

F:\MongoDB Tools\bin>
```

1. We are specifying the db option to say which database the data should be imported to
2. The type option is to specify that we are importing a csv file
3. Remember that the first row is called the header line which will become the Field names of the collection, that is why we specify the –headerline option. And then we specify our data.csv file.

# In Compass

MongoDB Compass - localhost:27017

Connect View Help

Local

4 DBS 2 COLLECTIONS

☆ FAVORITE

HOST  
localhost:27017

CLUSTER  
Standalone

EDITION  
MongoDB 5.0.2 Community

Filter your data

> TestDB

> admin

> config

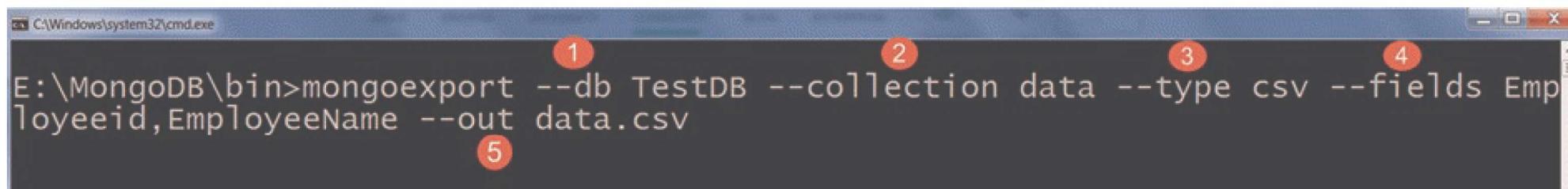
> local

Databases Performance

CREATE DATABASE

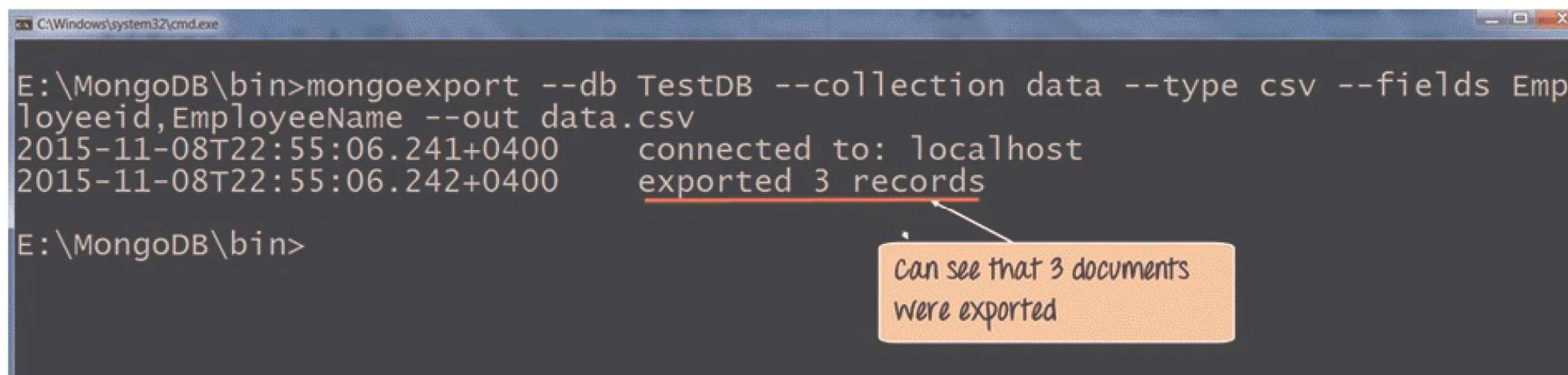
Database Name	Storage Size	Collections	Indexes
TestDB	20.0KB	1	1
admin	20.0KB	0	1
config	36.0KB	0	2
local	20.0KB	1	1

# Exporting MongoDB is done by using the mongoexport command



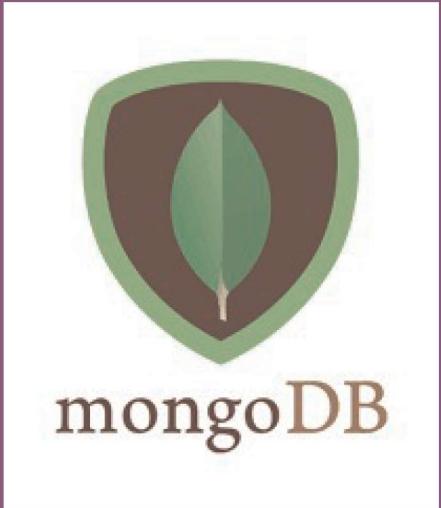
```
E:\MongoDB\bin>mongoexport --db TestDB --collection data --type csv --fields EmployeeId,EmployeeName --out data.csv
```

1. We are specifying the db option to say which database the data should be exported from.
2. We are specifying the collection option to say which collection to use
3. The third option is to specify that we want to export to a csv file
4. The fourth is to specify which fields of the collection should be exported.
5. The –out option specifies the name of the csv file to export the data to.



```
E:\MongoDB\bin>mongoexport --db TestDB --collection data --type csv --fields EmployeeId,EmployeeName --out data.csv
2015-11-08T22:55:06.241+0400      connected to: localhost
2015-11-08T22:55:06.242+0400      exported 3 records
```

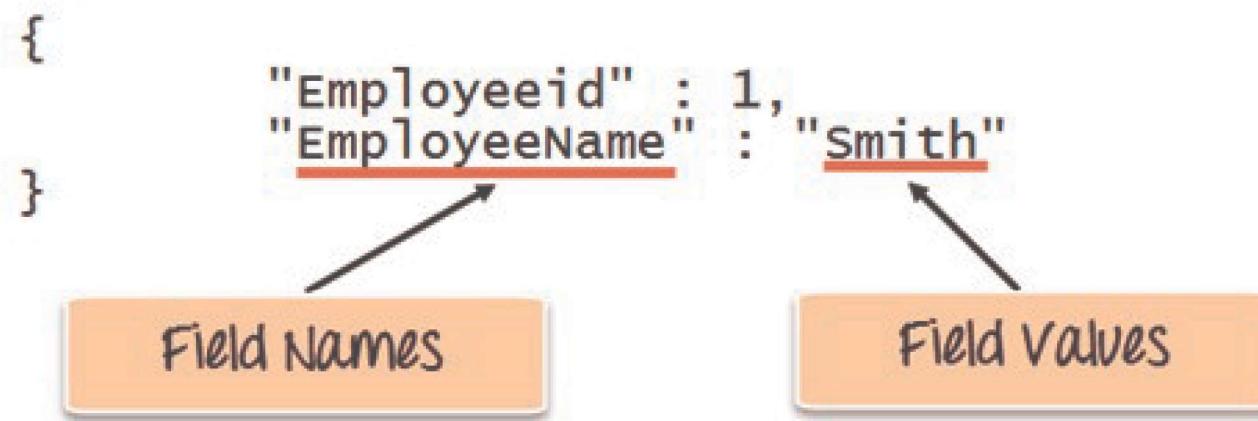
Can see that 3 documents  
were exported



# Create Database & Collection In MongoDB

## Introduction

- In MongoDB, the first basic step is to have a database and collection in place. The database is used to store all of the collections, and the collection in turn is used to store all of the documents. The documents in turn will contain the relevant Field Name and Field values.



- The Field Names of the document are “Employeeid” and “EmployeeName” and the Field values are “1” and “Smith’ respectively. A bunch of documents would then make up a collection in MongoDB.

## Use Database method:

- There is no create database command in MongoDB. Actually, MongoDB do not provide any command to create database.
- It may be look like a weird concept, if you are from traditional SQL background where you need to create a database, table and insert values in the table manually.
- Here, in MongoDB you don't need to create a database manually because MongoDB will create it automatically when you save the value into the defined collection at first time.

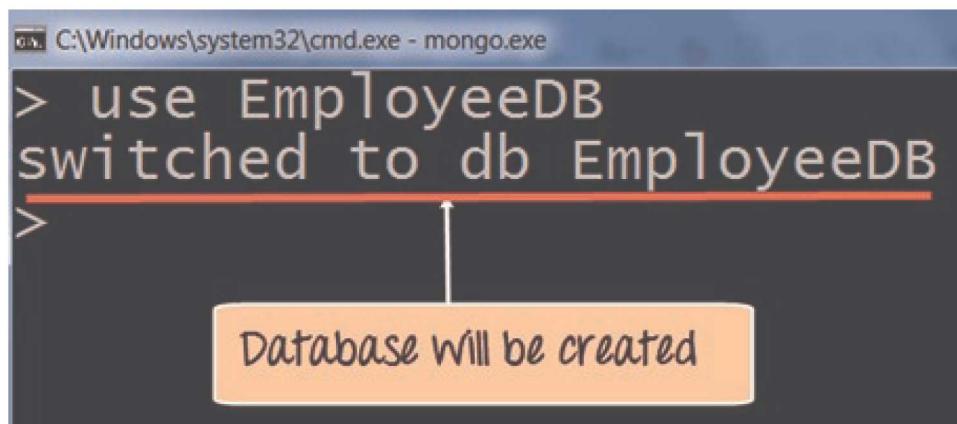
## Creating a database using “use” command

- Creating a database in MongoDB is as simple as issuing the “**using**” command



```
C:\Windows\system32\cmd.exe - mongo.exe
> use EmployeeDB
```

- The “**use**” command is used to create a database in MongoDB. If the database does not exist a new one will be created. If the database already exists, it will return the existing database.



```
C:\Windows\system32\cmd.exe - mongo.exe
> use EmployeeDB
switched to db EmployeeDB
>
```

Cont ...

1. To check the currently selected database, use the command db:



```
C:\WINDOWS\system32\cmd.exe - mongo
> use EmployeeDB
switched to db EmployeeDB
> db
EmployeeDB
>
```

A screenshot of a Windows Command Prompt window titled "C:\WINDOWS\system32\cmd.exe - mongo". Inside the window, a MongoDB shell session is running. The user has run the "use EmployeeDB" command, which has switched the database to "EmployeeDB". Then, the "db" command was run, displaying the name "EmployeeDB" again. A final greater-than sign (>) is shown at the bottom, indicating the prompt for the next command.

2. To check the database list, use the command show dbs:



```
C:\WINDOWS\system32\cmd.exe - mongo
> show dbs
TestDB 0.000GB
admin 0.000GB
config 0.000GB
local 0.000GB
>
```

A screenshot of a Windows Command Prompt window titled "C:\WINDOWS\system32\cmd.exe - mongo". Inside the window, a MongoDB shell session is running. The user has run the "show dbs" command, which lists four databases: "TestDB", "admin", "config", and "local", each with a size of "0.000GB". A final greater-than sign (>) is shown at the bottom, indicating the prompt for the next command.

- Here, our created database “EmployeeDB” is not present in the list, **insert at least one document** into it to display database

```
db.Employee.insert
(
    {
        "Employeeid" : 1,
        "EmployeeName" : "Martin"
    }
)
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> db.Employee.insert({"Employeeid" : 1,"EmployeeName" : "Martin"})
WriteResult({ "nInserted" : 1 })
>
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> db.Employee.insert({"Employeeid" : 1,"EmployeeName" : "Martin"})
WriteResult({ "nInserted" : 1 })
> show dbs
EmployeeDB 0.000GB
TestDB 0.000GB
admin 0.000GB
config 0.000GB
local 0.000GB
test 0.000GB
>
```

MongoDB Compass - localhost:27017

Connect View Help

Local

4 DBS 2 COLLECTIONS

CREATE DATABASE

Database Name	Storage Size	Collections	Indexes	
EmployeeDB	4.0KB	1	1	
TestDB	20.0KB	1	1	
admin	20.0KB	0	1	

HOST  
localhost:27017

CLUSTER  
Standalone

EDITION  
MongoDB 5.0.2 Community

Filter your data

# MongoDB Drop Database

- The dropDatabase command is used to drop a database. It also deletes the associated data files. It operates on the current database.

```
db.dropDatabase()
```

- This syntax will delete the selected database. In the case you have not selected any database, it will delete default "test" database.

- To check the database list, use the command show dbs:

```
show dbs
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> show dbs
EmployeeDB  0.000GB
TestDB      0.000GB
admin       0.000GB
config      0.000GB
local       0.000GB
test        0.000GB
>
```

- If you want to delete the database “TestDB”, use the dropDatabase() command as follows:

```
use TestDB
db.dropDatabase()
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> use TestDB
switched to db TestDB
> db.dropDatabase()
{ "ok" : 1 }
>
```

## Creating a Collection/Table using insert()

- The easiest way to create a collection is to insert a record (which is nothing but a document consisting of Field names and Values) into a collection. If the collection does not exist a new one will be created.

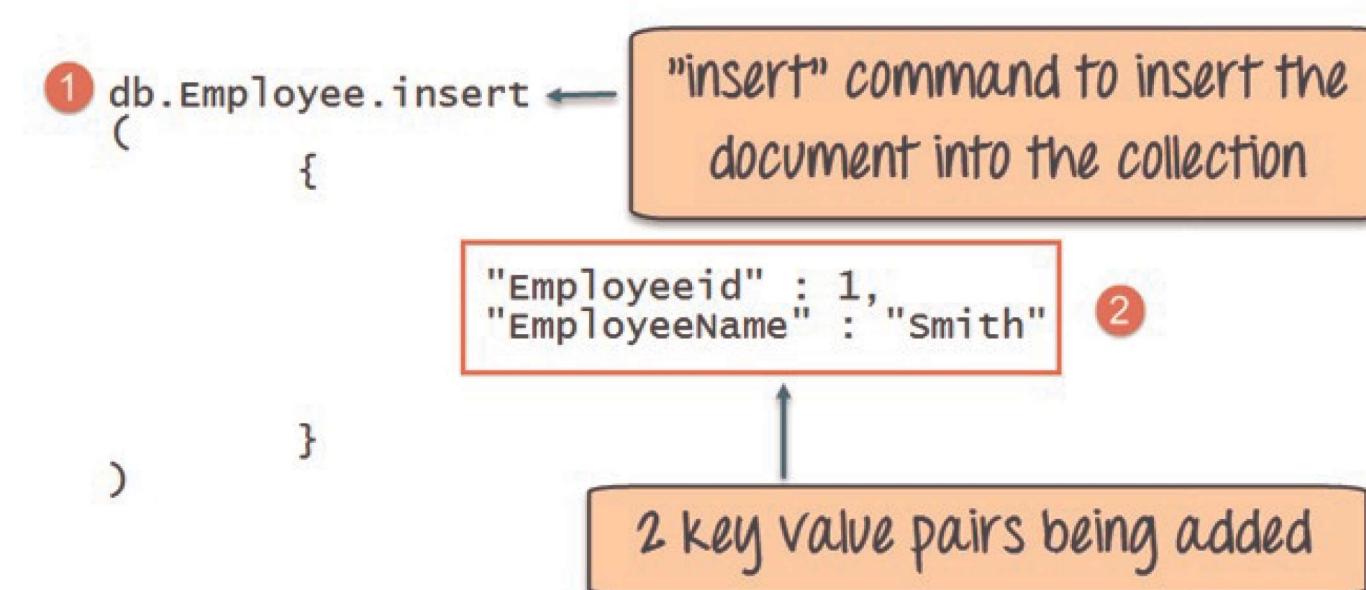
```
db.Employee.insert
(
    {
        "Employeeid" : 1,
        "EmployeeName" : "Martin"
    }
)
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> db.Employee.insert({ "Employeeid" : 1, "EmployeeName" : "Martin" })
WriteResult({ "nInserted" : 1 })
>
```

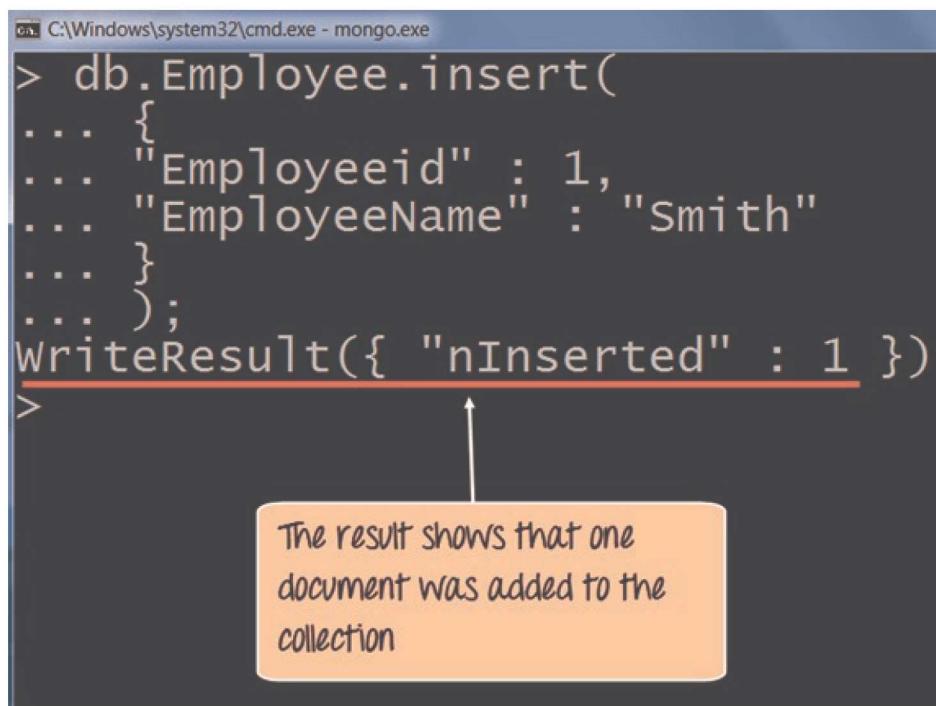
- By using the “**insert**” command the collection will be created.

## Adding documents using insert() command

- MongoDB provides the **insert () command** to insert documents into a collection. The following example shows how this can be done.
  1. Write the “insert” command
  2. Within the “insert” command, add the required Field Name and Field Value for the document which needs to be created.



1. The first part of the command is the “**insert statement**” which is the statement used to insert a document into the collection.
2. The second part of the statement is to add the Field name and the Field value, in other words, what is the document in the collection going to contain.



The screenshot shows a Windows Command Prompt window titled "C:\Windows\system32\cmd.exe - mongo.exe". Inside, a MongoDB shell command is run:

```
> db.Employee.insert(
... {
...   "Employeeid" : 1,
...   "EmployeeName" : "Smith"
... }
... );
WriteResult({ "nInserted" : 1 })
```

An orange callout box with a white border and rounded corners is positioned below the output. It contains the text: "The result shows that one document was added to the collection". An upward-pointing arrow originates from the bottom edge of the callout box and points to the "nInserted" field in the output line.



# MongoDB Datatypes

# Datatypes

Data Types	Description
String	String is the most commonly used datatype. It is used to store data. A string must be UTF 8 valid in mongodb.
Integer	Integer is used to store the numeric value. It can be 32 bit or 64 bit depending on the server you are using.
Boolean	This datatype is used to store boolean values. It just shows YES/NO values.
Double	Double datatype stores floating point values.
Min/Max Keys	This datatype compare a value against the lowest and highest bson elements.
Arrays	This datatype is used to store a list or multiple values into a single key.

## Cont ...

Arrays	This datatype is used to store a list or multiple values into a single key.
Object	Object datatype is used for embedded documents.
Null	It is used to store null values.
Symbol	It is generally used for languages that use a specific type.
Date	This datatype stores the current date or time in unix time format. It makes you possible to specify your own date time by creating object of date and pass the value of date, month, year into it.



# Creating and Dropping Collections

# MongoDB Create Collection

- In MongoDB, `db.createCollection(name, options)` is used to create collection. But usually you don't need to create collection. MongoDB creates collection automatically when you insert some documents.
- How to create collection:

```
db.createCollection (Name , Options)
```

- **Name:** is a string type, specifies the name of the collection to be created.
- **Options:** is a document type, specifies the memory size and indexing of the collection. It is an optional parameter.

# list of options

Field	Type	Description
Capped	Boolean	(Optional) If it is set to true, enables a capped collection. Capped collection is a fixed size collection that automatically overwrites its oldest entries when it reaches its maximum size. If you specify true, you need to specify size parameter also.
AutoIndexID	Boolean	(Optional) If it is set to true, automatically create index on ID field. Its default value is false.
Size	Number	(Optional) It specifies a maximum size in bytes for a capped collection. If capped is true, then you need to specify this field also.
Max	Number	(Optional) It specifies the maximum number of documents allowed in the capped collection.

```
use EmployeeDB
```

```
C:\WINDOWS\system32\cmd.exe - mongo
```

```
> use EmployeeDB
switched to db EmployeeDB
>
```

```
db.createCollection("Departments")
```

```
C:\WINDOWS\system32\cmd.exe - mongo
```

```
> use EmployeeDB
switched to db EmployeeDB
> db.createCollection("DEPARTMENTS")
{ "ok" : 1 }
>
```

# Inserting Documents

```
C:\WINDOWS\system32\cmd.exe - mongo
> db.Departments.insert(
... {
... "DepartmentId" : 10,
... "DepartmentName" : "Administration",
... "LocationId" : 1700,
... "ManagerId" : 100
... }
... )
WriteResult({ "nInserted" : 1 })
>
```

# Showing Collections and Viewing Inserted Documents

```
show collections
```

```
C:\WINDOWS\system32\cmd.exe - mongo
```

```
> show collections
```

```
Departments
```

```
Employee
```

```
>
```

- If you want to see the inserted document, use the find() command.

```
db.Collection_Name.find()
```

```
C:\WINDOWS\system32\cmd.exe - mongo
```

```
> db.Departments.find()
```

```
{ "_id" : ObjectId("613271480bc1fda7d576ed0d"), "DepartmentId" : 10, "DepartmentName" : "Administration", "LocationId" : 1700, "ManagerId" : 100 }
```

```
>
```

## MongoDB Drop collection

- In MongoDB, db.collection.drop() method is used to drop a collection from a database. It completely removes a collection from the database and does not leave any indexes associated with the dropped collections.
- The db.collection.drop() method does not take any argument and produce an error when it is called with an argument. This method removes all the indexes associated with the dropped collection.

```
db.CollectionName.drop()
```

# First check the already existing collections in your database.

```
use EmployeeDB
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> use EmployeeDB
switched to db EmployeeDB
>
```

```
show collections
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> show collections
Departments
Employee
>
```

# Dropping a Collection

```
db.collections.drop()
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> show collections
Departments
Employee
> db.Departments.drop()
true
>
```

```
show collections
```

```
C:\WINDOWS\system32\cmd.exe - mongo
> show collections
Departments
Employee
> db.Departments.drop()
true
> show collections
Employee
>
```

# Summary

In this lesson, you should have learned how to:

- Installation of MongoDB and MongoDB Compass
- Working with MongoShell
- Configuration
- Collections and Documents
- Data Types

