



TASK

Database Interaction

Visit our website

Introduction

WELCOME TO THE DATABASE INTERACTION TASK!

Now that you have set up your MongoDB Cluster using Atlas, we are going to learn how to use Mongo (MongoDB's administrative shell) to create databases and collections. You will also learn to create, read, update, and delete documents from collections.



Get in touch
Connect for support

Remember that with our courses, you're not alone! You can contact an expert code reviewer to get support on any aspect of your course.

The best way to get help is to login to Discord at <https://discord.com/invite/hyperdev> where our specialist team is ready to support you.

Our team is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

MONGO SHELL BASIC COMMANDS

In the previous task, you created a MongoDB database. In this task, we are going to learn how to manipulate our database using CRUD operations. CRUD is simply an acronym for the 4 basic operations used to manipulate a database: Create, Read, Update, and Delete.

Before considering CRUD though, you first need to learn how to use some basic mongo shell commands:

- **show dbs;**

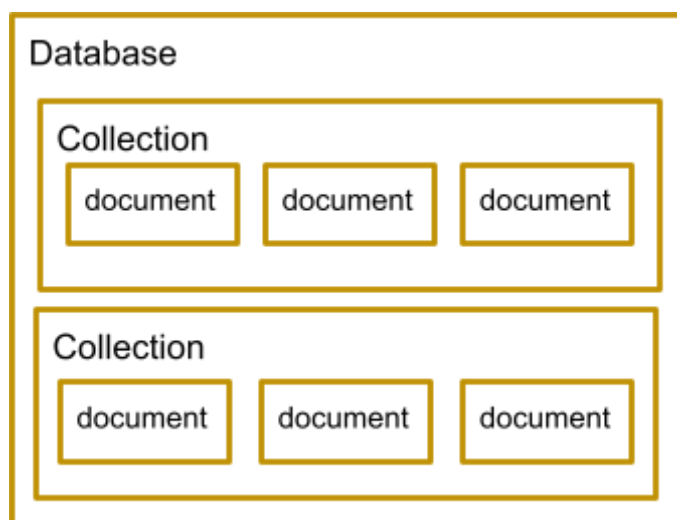
This command is used to list all the databases in your cluster.

- **use db_name;**

This command is used to select a database. Before you can manipulate a database, you need to select it. This command is also used to create a database (as you did in your previous task). If the database does not already exist, this instruction will create a database with the name specified in the command. The rules regarding which special characters are not to be used in naming your database vary depending on the operating system your database is running on. You should not use the following characters in your db name: \. "\$*<>:|?

- **show collections;**

This command shows all the collections in the previously selected database. A collection is a grouping of related documents and is equivalent to a table in a relational database. If you were creating a database for a company that sells cars, for example, you could have a car collection, customer collection, and shop collection.



- **db.dropDatabase();**

This command is used to delete a database. Before you use this command though, you need to make sure you have selected the database you would like to delete using the 'use' command shown previously. If you are not sure which database you are currently working with, you can simply type the command 'db' into your mongo shell and it will display the name of the database that you have selected.

CRUD OPERATIONS

Create:

Once you have created a database, you need to insert collections and documents into the database. As you know, a collection is a grouping of BSON documents. A BSON document is basically a JSON document that is stored in such a way that it allows you to store more 'type' information about your data. For example, JSON files don't recognise 'date' as a type, whereas BSON documents do. As you can see in the image below, documents store information using key-value pairs. Each document in a collection can have a slightly different structure.

```
{  
  name: "sue",  
  age: 26,  
  status: "A",  
  groups: [ "news", "sports" ]  
}
```



Image source: <https://docs.mongodb.com/manual/core/document/>

To insert a new document into a collection, use the **insertOne** or **insertMany** command as shown below:

```
db.people.insertOne({name: 'Sue', age: 33});  
or  
db.people.insertMany([  
  ↑      ↑      ↑  
  {name: 'Sue', age: 33}, {name: 'Sam', surname: 'Deans', age: 25}]);  
collection document do
```

Note that not all the documents in your collection need exactly the same format. See how the BSON document for Sue only stores name and age, whereas the BSON document for Sam stores name, surname, and age information. If the collection you specify in the **insertOne** or **insertMany** command does not exist, MongoDB will create the collection for you. Every document needs a unique identification. In MongoDB the field `_id` must be specified for each document. If you don't explicitly specify an `_id` field when you insert a document, MongoDB will automatically generate one for you.

Read:

Once your collection contains documents, you will want to read those documents. We do this using the **find()** command as illustrated below.

```
db.people.find().pretty();
```

Or

```
db.people.find({name: 'Tom'});
```

Or

```
db.people.find({name: 'Sue'}, {_id: false, age:true})
```



collection

The **pretty()** method is used to make the output more readable. The **find** command can be used without it.

When you specify **find** without any arguments, all documents in the collection will be returned.

You can also pass one or more key-value pairs as shown in the second code example, which will then find all documents that match the criteria specified. E.g. in the second code example above, the **find** command will find all documents where the person has the name, Tom, as well as all the information contained in all those documents, including the `_id` for each document.

You often do not want to read or display all the information in a document. For example, you may not want a user to see the `_id` value for each document. You can, therefore, also pass the **find()** method a second argument set as shown in

the third example above. The arguments in the second set of curly brackets (`{_id: false, age: true }`) specify which fields will be retrieved and which won't be. In this example, the `_id` won't be output but the `age` field will.

You can also use the `findOne()` method instead of the `find()` method. `findOne()` will return only the first document that matches the specified criteria.

Update:

The `update()` command is used to update documents in a collection.

```
db.people.update({name: 'Sue'}, {age: 34, name: 'Sue'})

// Or

db.people.update({name: 'Sue'}, {$set: {age: 34}})

// Or

db.people.update({name: 'Sue'}, {$set: {name: 'Susan'}}, {multi: true})
```

Compare the first two commands in the code above. If you specify **update** without using the `$set` keyword, the entire document will be updated (as in the first command). However, if you use the `$set` keyword only the field specified will be updated and the other fields will remain the same. If you pass the third argument `{multi: true}` (as in the third command), then all documents that match the specified criteria will be updated. With the latest versions of MongoDB, you can use `updateOne()` and `updateMany()` commands. See the recommended reading for this task for more on this and other commands.

Be careful when you use the update command! Fields not defined in the update section will be removed from the document. Similarly, if you specify fields that were not previously part of your document with the update command, new fields will be added to your document.

Delete:

To delete documents from a collection use the `remove()` command as shown in the code examples below.

```
db.people.remove();

// Or

db.people.remove({name: 'Sue'})

// Or

db.people.remove({name: 'Sue'}, true)
```

If you issue the **remove()** command without passing any arguments (as in the first example), all documents in the collection will be removed. If you pass one argument (as in the second example) all documents with the specified criteria will be removed. If you pass the value **true** as a second argument (as in the third example) only the first document that meets the specified criteria will be removed. Instead of using the **remove** command, you can also use the **deleteOne()** or **deleteMany()** command with the latest versions of MongoDB. See the recommended reading for this task for more on this and other commands.



Extra resource

For more information about performing CRUD operations with MongoDB, see chapters 2 - 6 of the additional reading that accompanies this task. This e-book is written and made available by the 'beautiful people of Stack Overflow'.

SPOT CHECK 1

Let's see what you can remember from this section.

1. What characters should not be used in the name of your database?
2. What is the difference between a JSON and a BSON document?
3. What are the two commands you can use to read documents?
4. What does the **\$set** keyword do?

Compulsory Task 1

Follow these steps:

- Create a document called **taskCRUD**. Add screenshots of your CLI that provide evidence of how you have successfully carried out every instruction specified in this task.
- In the database you created in your previous task, do the following using the mongo shell:
 - Add a collection called 'cars' and add at least 5 documents to your collection. Assume that you will have some of the following information for each car you add to your collection: The model, make, colour, registration number, owner, address.
 - Include another document that contains the following information:
 - Model: 2005
 - Make: Ford Fiesta
 - Owner: Sue Bailey
 - Registration: ABC 123 GP
 - Address: 13 Main Road, Johannesburg, South Africa.
 - Once you have added all the cars to your database, display all the documents you have added.
 - Assume that Sue Bailey moves. Update her address to 21 Maureen Street, Bluewater Bay, Port Elizabeth, South Africa.
 - Now assume that Sue marries and her surname changes to Smith. Update your database accordingly.
 - Display all cars older than 5 years.
 - Create another document where there is another Ford Fiesta owned by someone called Sue Smith.
 - Our original Sue Smith (

- née Bailey) has her car scrapped. Remove the document that described her car from your database. Be sure to remove only her document!
- Imagine that you would like to store all the previous owners for a specific car. Add a document to a database that does this. This car should have the names of at least 3 previous owners.

If you are having any difficulties, please feel free to contact our specialist team [on Discord](#) for support.

Completed the task(s)?

Ask an expert to review your work!

[Review work](#)



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.



SPOT CHECK 1 ANSWERS

1. `\. "$*<>:|?`
2. A BSON document is a JSON document that is stored in such a way that it allows you to store more 'type' information about your data.
3. `find()` and `findOne()`
4. If you specify **update** without using the **\$set** keyword, the entire document will be updated. However, if you use the **\$set** keyword only the field specified will be updated and the other fields will remain the same.