Creating a Database

- Open the command prompt and navigate to the **/bin** folder of the MongoDB using the cd command and execute the command mongod there. This will initiate the MongoDB server. We have to keep this command prompt window alive, as this is running MongoDB. To stop the MongoDB server, simply enterexit and press Enter.
- Now, Open another command prompt and navigate to the **/bin** folder of the MongoDB again and execute the command mongo. This will open up the client to run the MongoDB commands.

Creating a Database

- **use *database_name***
- To check the current connected database, Use Command
  - **db**
- To see the list of all the databases in MongoDB, use command
  - **show dbs**
- Please note that the newly created dstabase **mynewdatabase** has not been listed after running the above command. This is because, no records have been inserted into that database yet. Just insert one record and then run the command again as shown below:

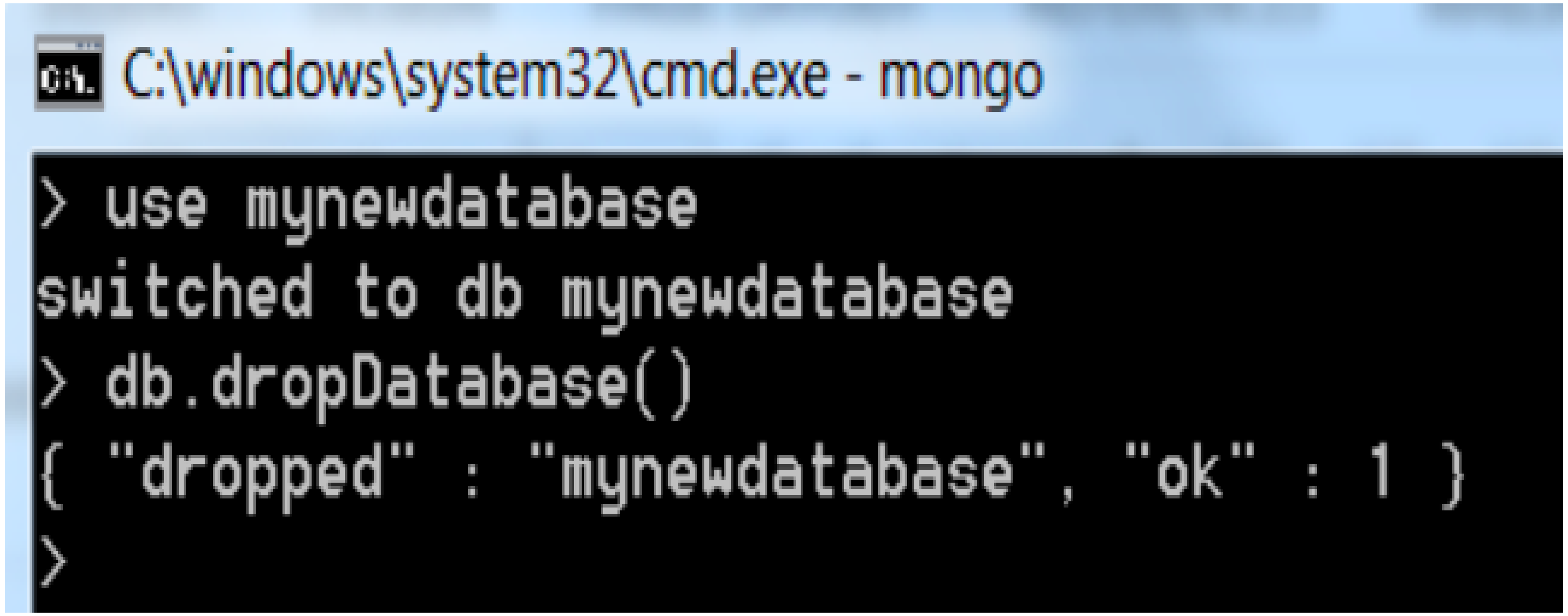
Creating a Database

- To Insert data, run the following command.
- db.student.insert({ **name** : "*Viraj*" })**NOTE :** In MongoDB, test will be the default database. If no database is created, then all the data will be stored in the test database.
- **Drop a Database**
- First check the list of databases available as shown below, using the show dbs command.

Creating a Database

- To Insert data, run the following command.
- db.student.insert({ **name** : "*Viraj*" })**NOTE :** In MongoDB, test will be the default database. If no database is created, then all the data will be stored in the test database.
- **Drop a Database**
- First check the list of databases available as shown below, using the show dbs command.

Creating a Database



```
> use mynewdatabase
switched to db mynewdatabase
> db.dropDatabase()
{ "dropped" : "mynewdatabase", "ok" : 1 }
>
```

Creating a Database

- **Creating a Collection**
- In MongoDB a collection is automatically created when it is referenced in any command. For example, if you run an insert command :

db.**student**.*insert*({

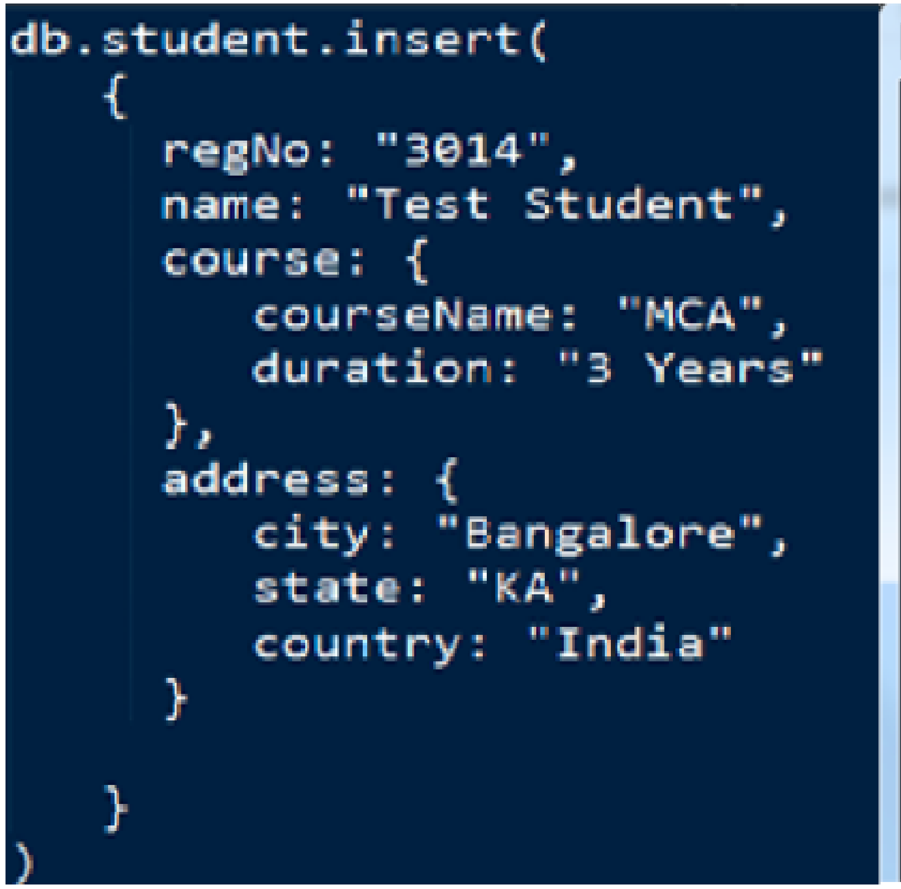
name: "*Viraj*“

})

Creating a Database

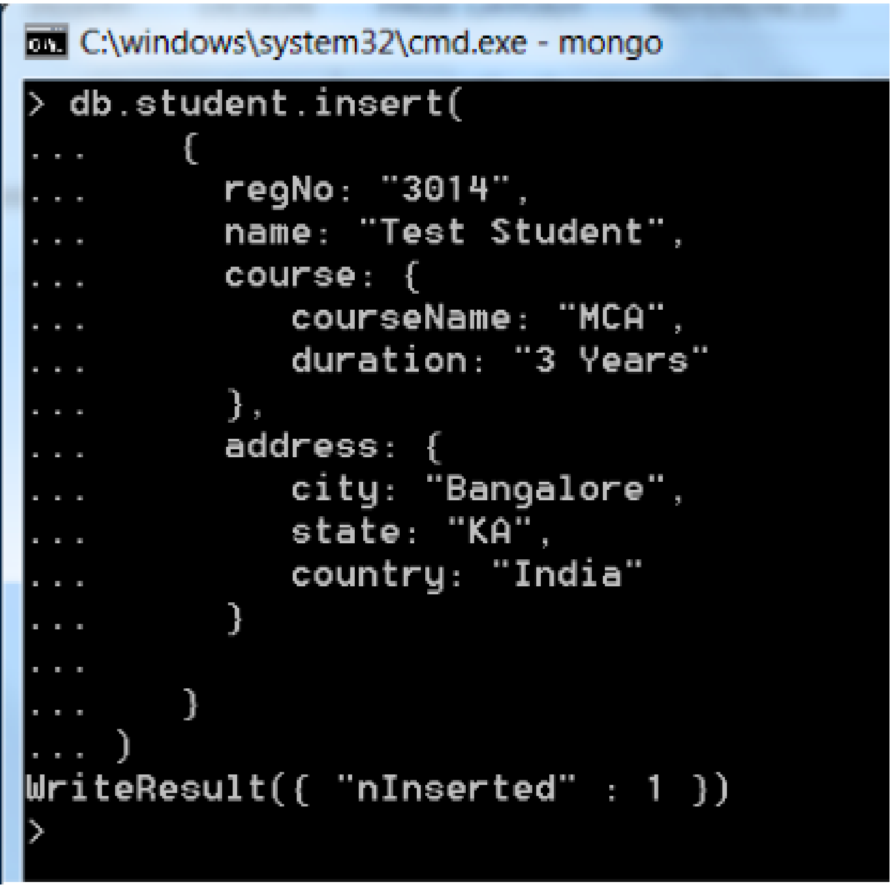
- **Drop a Collection**
- Any collection in a database in MongoDB can be dropped easily using the following command :
  - db.**collection_name**.*drop*()
- drop() method will return true is the collection is dropped successfully, else it will return false.

Creating a Database



```
db.student.insert(
    {
      regNo: "3014",
      name: "Test Student",
      course: {
          courseName: "MCA",
          duration: "3 Years"
      },
      address: {
          city: "Bangalore",
          state: "KA",
          country: "India"
      }

    }
)
```



```
> db.student.insert(
...     {
...         regNo: "3014",
...         name: "Test Student",
...         course: {
...             courseName: "MCA",
...             duration: "3 Years"
...         },
...         address: {
...             city: "Bangalore",
...             state: "KA",
...             country: "India"
...         }
...
...     }
... )
WriteResult({ "nInserted" : 1 })
>
```

Creating a Database

- **Updating a document in a collection (Update)**
- In order to update specific field values of a collection in MongoDB, run the below query.
  - db.**collection_name**.*update*()
- update() method specified above will take the fieldname and the new value as argument to update a document.

Creating a Database

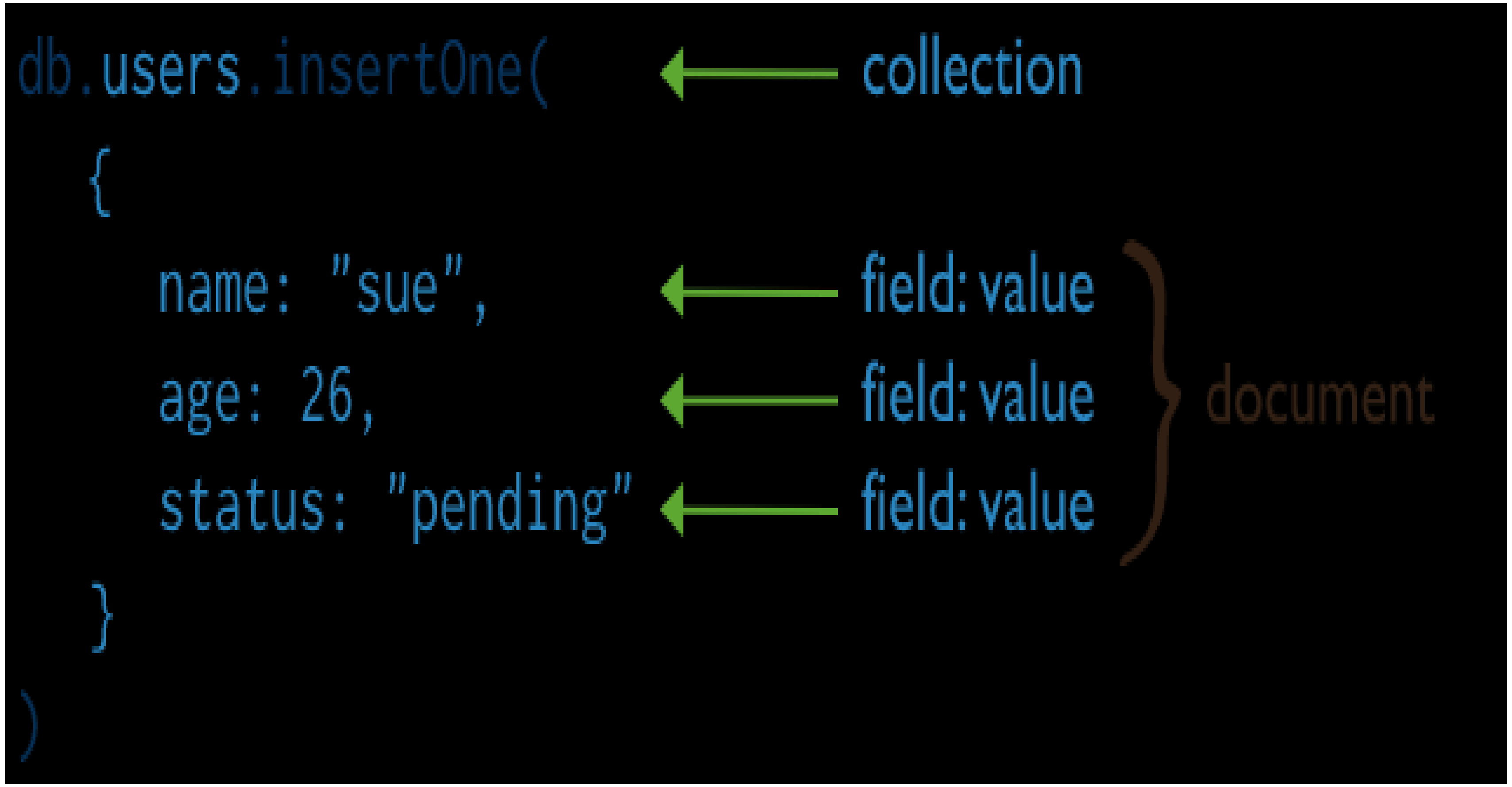
- **Removing an entry from the collection (Delete)**
- Let us now look into the deleting an entry from a collection. In order to delete an entry from a collection, run the command as shown below :
  - db.**collection_name**.*remove*({"**fieldname**":"*value*"})
  - For Example : db.student.remove({"regNo":"3014"})

MongoDB CRUD Operations

- **Create Operations**
- Create or insert operations add new documents to a collection. If the collection does not currently exist, insert operations will create the collection.
- MongoDB provides the following methods to insert documents into a collection:
  - db.collection.insertOne()
  - db.collection.insertMany()

  - In MongoDB, insert operations target a single collection. All write operations in MongoDB are atomic on the level of a single document.

MongoDB CRUD Operations

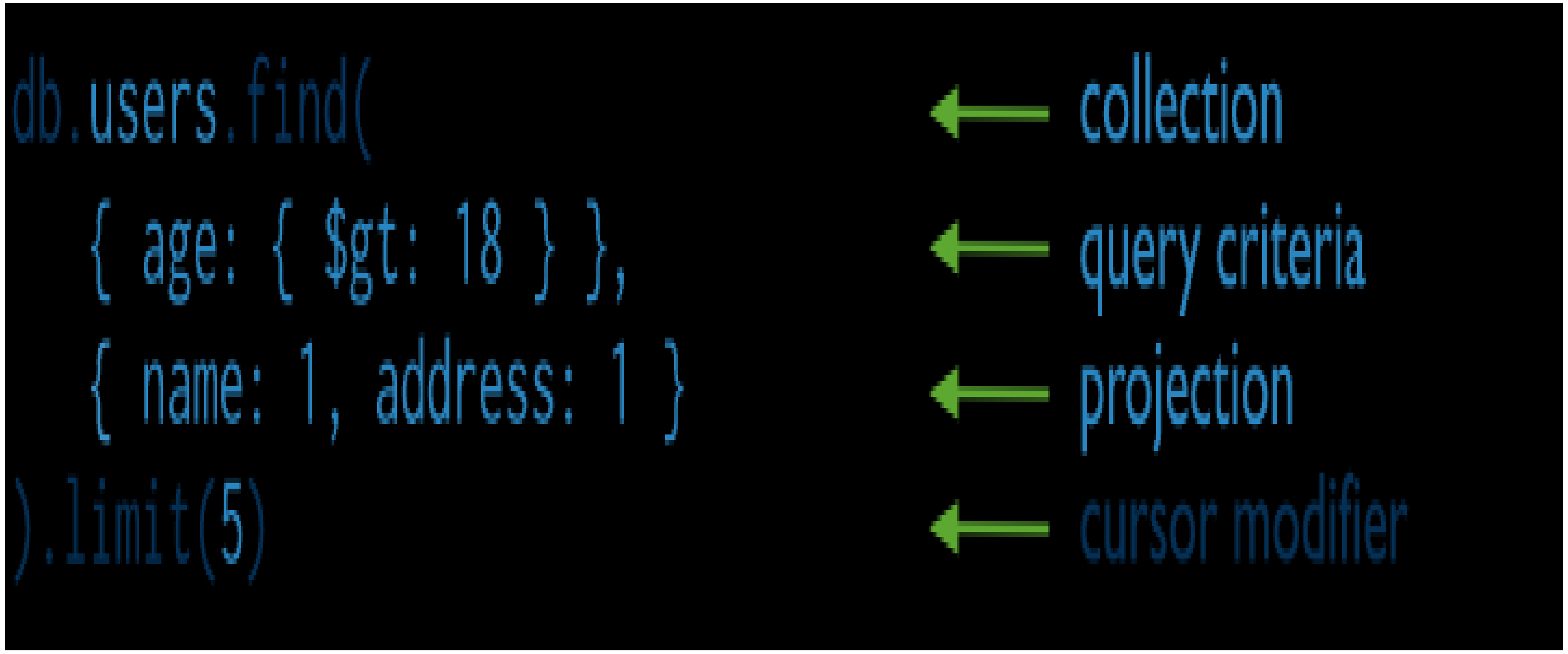


```
db.users.insertOne(
  {
    name: "sue",
    age: 26,
    status: "pending"
  }
)
```

MongoDB CRUD Operations

- **Read Operations**
- Read operations retrieves documents from a collection; i.e. queries a collection for documents. MongoDB provides the following methods to read documents from a collection:
  - db.collection.find()
- You can specify query filters or criteria that identify the documents to return.

MongoDB CRUD Operations



```
db.users.find(
   { age: { $gt: 18 } },
   { name: 1, address: 1 }
).limit(5)
```

MongoDB CRUD Operations

```
db.inventory.insertMany([
   { item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" },
   { item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "A" },
   { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },
   { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },
   { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" }
]);
```

MongoDB CRUD Operations

- Select All Documents in a Collection
- To select all documents in the collection, pass an empty document as the query filter parameter to the find method. The query filter parameter determines the select criteria:

- db.inventory.find({})
- This operation corresponds to the following SQL statement:
- **SELECT * FROM** inventory

MongoDB CRUD Operations

- The following example selects from the inventory collection all documents where the status equals "D":
  - db.inventory.find({ status: "D" })
- This operation corresponds to the following SQL statement:
  - **SELECT** * **FROM** inventory **WHERE** status = "D"

MongoDB CRUD Operations

- The following example retrieves all documents from the inventory collection where status equals either "A" or "D":

  - db.inventory.find({ status: { $in: ["A", "D"] } })

NOTE

- Although you can express this query using the $or operator, use the $in operator rather than the $or operator when performing equality checks on the same field.
- The operation corresponds to the following SQL statement:
  - **SELECT * FROM** inventory **WHERE** status **in** ("A", "D")

MongoDB CRUD Operations

- Specify AND Conditions
- A compound query can specify conditions for more than one field in the collection’s documents. Implicitly, a logical AND conjunction connects the clauses of a compound query so that the query selects the documents in the collection that match all the conditions.
- The following example retrieves all documents in the inventory collection where the status equals "A"**and** qty is less than ($lt) 30:
  - db.inventory.find({ status: "A", qty: { $lt: 30 } })
- The operation corresponds to the following SQL statement:
  - **SELECT** * **FROM** inventory **WHERE** status = "A" **AND** qty < 30

MongoDB CRUD Operations

- **Specify AND Conditions**
- A compound query can specify conditions for more than one field in the collection’s documents. Implicitly, a logical AND conjunction connects the clauses of a compound query so that the query selects the documents in the collection that match all the conditions.
- The following example retrieves all documents in the inventory collection where the status equals "A"**and** qty is less than ($lt) 30:
  - db.inventory.find({ status: "A", qty: { $lt: 30 } })
- The operation corresponds to the following SQL statement:
  - **SELECT** * **FROM** inventory **WHERE** status = "A" **AND** qty < 30

MongoDB CRUD Operations

- **Specify AND as well as OR Conditions**
- In the following example, the compound query document selects all documents in the collection where the status equals "A" **and** *either* qty is less than ($lt) 30 *or* item starts with the character p:
  - db.inventory.find({ status: "A", $or: [{ qty: { $lt: 30 } }, { item: /^p/ }] })
- The operation corresponds to the following SQL statement:
  - **SELECT** * **FROM** inventory **WHERE** status = "A" **AND** (qty < 30 **OR** item **LIKE** "p%")