MONGODB: CRUDE

- Shankarganesh B

- Created a new database named library and switched to it.
- Added a collection called books.
- Inserted a few favorite books into the books collection, including details such as:
  title, author, and publishing year
- Used the find( ) method to fetch a book where the author's name is David.

```
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>mongosh
Current Mongosh Log ID: 675ec6657b19bd3efc893bf7
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.4
Using MongoDB:          8.0.4
Using Mongosh:          2.3.4
mongosh 2.3.6 is available for download: https://www.mongodb.com/try/download/shell

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

------
   The server generated these startup warnings when booting
   2024-12-13T02:16:02.483+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

test> use library
switched to db library
library> db.createCollection("books")
{ ok: 1 }
library> db.books.insertOne({title: "The Fury", author:"Alex",published_year:2024})
{
  acknowledged: true,
  insertedId: ObjectId('675ec7567b19bd3efc893bf8')
}
library> db.books.insertMany([{title: "The Man", author:"David", published_year:2021},{title:"The Right Time", author:"Pacito Marwin", published_year:2017}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('675ec84a7b19bd3efc893bf9'),
    '1': ObjectId('675ec84a7b19bd3efc893bfa')
  }
}
library> db.books.find({author:"David"})
[
  {
    _id: ObjectId('675ec84a7b19bd3efc893bf9'),
    title: 'The Man',
    author: 'David',
    published_year: 2021
  }
]
```

- Queried the **books** collection to fetch the earliest published book by identifying the record with the oldest **publishing year**.
- Retrieved all books from the collection and sorted them chronologically in ascending order (from the oldest to the newest) based on their **publishing year**.
- Updated the **publishing year** for the book titled **The Fury** to reflect the correct year.

```
library> db.books.find().sort({published_year:1}).limit(1)
[
  {
    _id: ObjectId('675ec84a7b19bd3efc893bfa'),
    title: 'The Right Time',
    author: 'Pacito Marwin',
    published_year: 2017
  }
]
library> db.books.find().sort({published_year:1})
[
  {
    _id: ObjectId('675ec84a7b19bd3efc893bfa'),
    title: 'The Right Time',
    author: 'Pacito Marwin',
    published_year: 2017
  },
  {
    _id: ObjectId('675e2e6c642ecb1dee893bf8'),
    title: 'More than words',
    author: 'Jill Santapole',
    published_year: 2020
  },
  {
    _id: ObjectId('675ec84a7b19bd3efc893bf9'),
    title: 'The Man',
    author: 'David',
    published_year: 2021
  },
  {
    _id: ObjectId('675ec7567b19bd3efc893bf8'),
    title: 'The Fury',
    author: 'Alex',
    published_year: 2024
  }
]
library> db.books.updateOne({title:"The Fury"},{$set:{published_year:2023}})
library>
```

- Updated the **genre** field to **Fiction** for all books in the **books** collection.
- Used the update method with the option multi: true to apply the change to multiple documents at once.

```
library> db.books.update({},{$set:{genre:"Fiction"}},{multi:true})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 4,
  modifiedCount: 4,
  upsertedCount: 0
}
library> db.books.find()
[
  {
    _id: ObjectId('675e2e6c642ecb1dee893bf8'),
    title: 'More than words',
    author: 'Jill Santapole',
    published_year: 2020,
    genre: 'Fiction'
  },
  {
    _id: ObjectId('675ec7567b19bd3efc893bf8'),
    title: 'The Fury',
    author: 'Alex',
    published_year: 2023,
    genre: 'Fiction'
  },
  {
    _id: ObjectId('675ec84a7b19bd3efc893bf9'),
    title: 'The Man',
    author: 'David',
    published_year: 2021,
    genre: 'Fiction'
  },
  {
    _id: ObjectId('675ec84a7b19bd3efc893bfa'),
    title: 'The Right Time',
    author: 'Pacito Marwin',
    published_year: 2017,
    genre: 'Fiction'
  }
]
```

- Used the deleteOne() method to remove a specific book with the title **The Fury** from the **books** collection. This operation targeted and deleted only the matching document.
- Used the remove() method with a condition {publishing_year: { $gt: 2017 } } to delete all books published after the year **2017**. This ensured that only books with a **publishing year** greater than **2017** were removed from the collection.

```
library> db.books.deleteOne({title:"The Fury"})
{ acknowledged: true, deletedCount: 1 }
library> db.books.find()
[
  {
    _id: ObjectId('675e2e6c642ecb1dee893bf8'),
    title: 'More than words',
    author: 'Jill Santapole',
    published_year: 2020,
    genre: 'Fiction'
  },
  {
    _id: ObjectId('675ec84a7b19bd3efc893bf9'),
    title: 'The Man',
    author: 'David',
    published_year: 2021,
    genre: 'Fiction'
  },
  {
    _id: ObjectId('675ec84a7b19bd3efc893bfa'),
    title: 'The Right Time',
    author: 'Pacito Marwin',
    published_year: 2017,
    genre: 'Fiction'
  }
]
library> db.books.remove({published_year:{$gt:2017}})
DeprecationWarning: Collection.remove() is deprecated. Use deleteOne, deleteMany, findOneAndDelete, or bulkWrite.
{ acknowledged: true, deletedCount: 2 }
library> db.books.find()
[
  {
    _id: ObjectId('675ec84a7b19bd3efc893bfa'),
    title: 'The Right Time',
    author: 'Pacito Marwin',
    published_year: 2017,
    genre: 'Fiction'
  }
]
```