Mongo DB – Hands on assignment

A) Inventory Management System

Q1) Create a database called Inventory.

-> use Inventory

Q2) Create a collection called Products with attributes: ProductID, Name, Category, Stock, and Price.

-> db.createCollection("Products")

Q3) Insert 10 product documents into the Products collection.

-> db.Products.insertMany([{ ProductID: 1, Name: "TV", Category: "Electronics", Stock: 7, Price: 10 }, { ProductID: 2, Name: "Mixer", Category: "Home Appliances", Stock: 32, Price: 12 }, { ProductID: 3, Name: "Utensils", Category: "Home Appliances", Stock: 35, Price: 9 }, { ProductID: 4, Name: "Fridge", Category: "Electronics", Stock: 0, Price: 15}, { ProductID: 5, Name: "Washing Machine", Category: "Electronics", Stock: 21, Price: 20 }, { ProductID: 6, Name: "Door", Category: "Furniture", Stock: 8, Price: 25 }, { ProductID: 7, Name: "Sofa", Category: "Furniture", Stock: 7, Price: 9}, { ProductID: 8, Name: "Jeans", Category: "Clothing", Stock: 15, Price: 16 }, { ProductID: 9, Name: "Shoes", Category: "Footwear", Stock: 5, Price: 13 }, {ProductID: 10, Name: "Sandals", Category: "Footwear", Stock: 13, Price: 12}])

-> Output:


```
[
  {
    _id: ObjectId('675af2b8235115c5a40d8190'),
    ProductID: 1,
    Name: 'TV',
    Category: 'Electronics',
    Stock: 7,
    Price: 10
  },
  {
    _id: ObjectId('675af2b8235115c5a40d8191'),
    ProductID: 2,
    Name: 'Mixer',
    Category: 'Home Appliances',
    Stock: 32,
    Price: 12
  },
  {
    _id: ObjectId('675af2b8235115c5a40d8192'),
    ProductID: 3,
    Name: 'Utensils',
    Category: 'Home Appliances',
    Stock: 35,
    Price: 9
  },
  {
    _id: ObjectId('675af2b8235115c5a40d8193'),
    ProductID: 4,
    Name: 'Fridge',
    Category: 'Electronics',
    Stock: 0,
    Price: 15
  },
  {
    _id: ObjectId('675af2b8235115c5a40d8194'),
    ProductID: 5,
    Name: 'Washing Machine',
    Category: 'Electronics',
    Stock: 21,
    Price: 20
  },
```

Q4) Retrieve all products with stock greater than 10.

-> db.Products.find({Stock:{$gt:10}})

-> Output:

Q5) Find all products from the 'Electronics' category

-> db.Products.find({Category: "Electronics"})

-> Output:

Q6) Update the price of the product with ProductID 7 to 19.99.

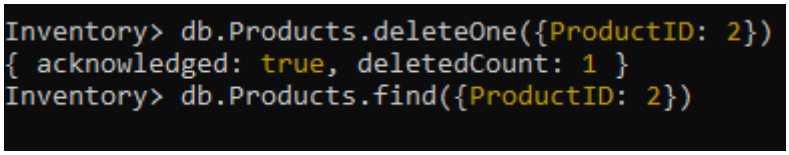
-> db.Products.updateOne({ProductID: 7}, {$set:{Price:19.99}})

-> Output:

Q7) Delete the product with ProductID 2.

-> db.Products.deleteOne({ProductID: 2})

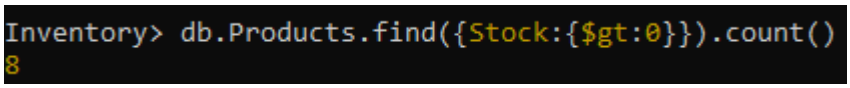
-> Output:



Q8) Count how many products are in stock.

-> db.Products.find({Stock:{$gt:0}}).count()

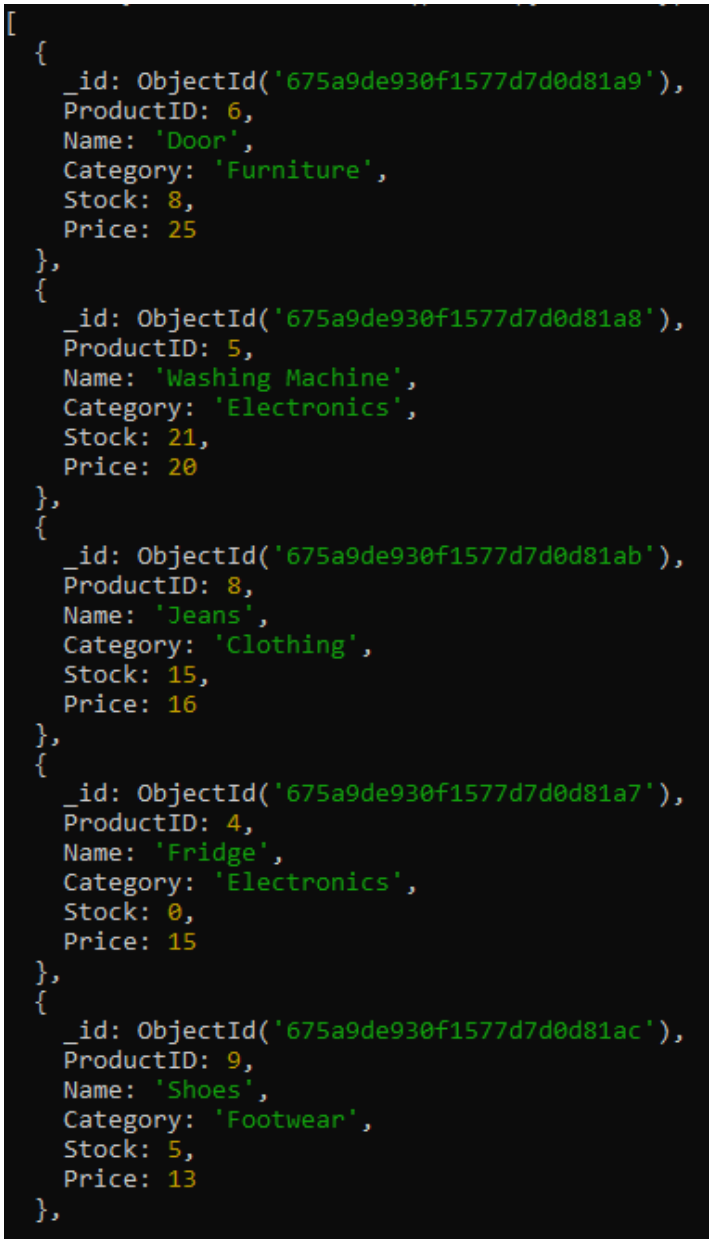
-> Output:



Q9) Sort products by Price in descending order.

-> db.Products.find().sort({Price:-1})

-> Output:



Q10) Find products whose price is either 10 or 15.

-> db.Products.find({Price:{$in:[10,15]}})

-> Output:

```
Inventory> db.Products.find({Price:{$in:[10,15]}})
[
  {
    _id: ObjectId('675a9de930f1577d7d0d81a4'),
    ProductID: 1,
    Name: 'TV',
    Category: 'Electronics',
    Stock: 7,
    Price: 10
  },
  {
    _id: ObjectId('675a9de930f1577d7d0d81a7'),
    ProductID: 4,
    Name: 'Fridge',
    Category: 'Electronics',
    Stock: 0,
    Price: 15
  }
]
```

Q11) Retrieve products whose Category is 'Home Appliances' and Stock is greater than 30.

-> db.Products.find({Category: "Home Appliances", Stock:{$gt:30}})

-> Output:

Q12) Update the Category of the product with ProductID 4 to 'Furniture'.

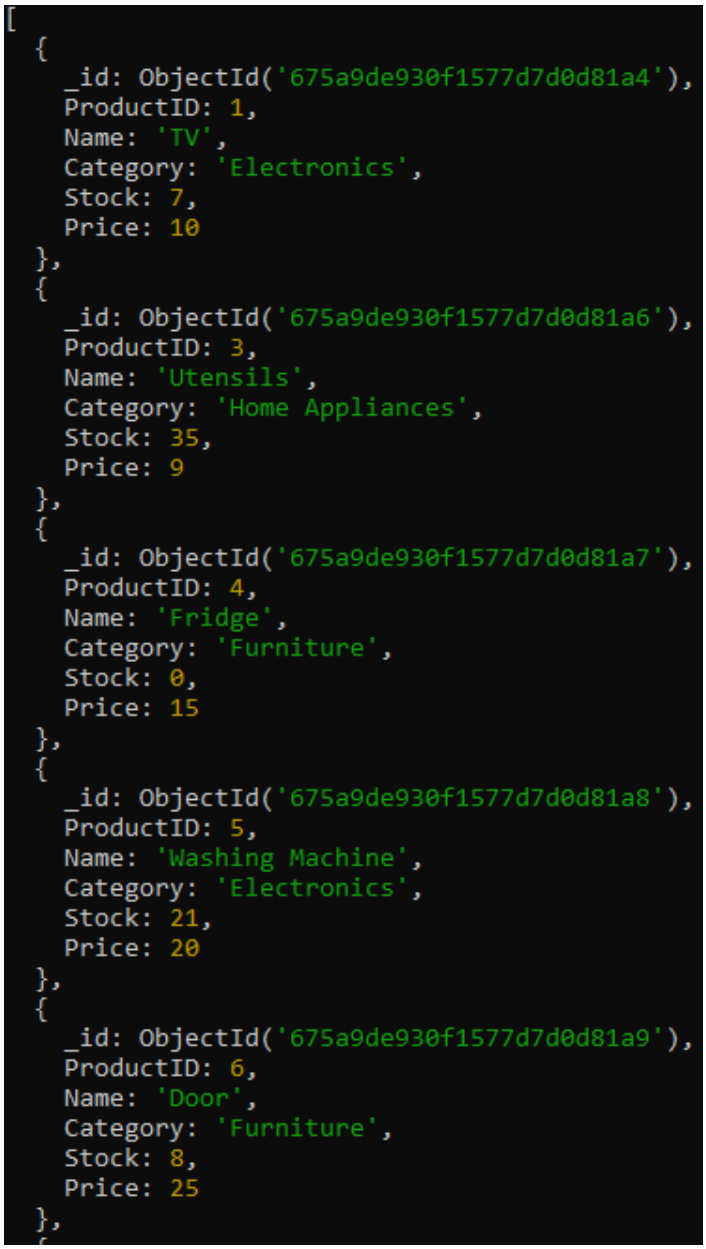
-> db.Products.updateOne({ProductID: 4}, {$set:{Category: "Furniture"}})

-> Output:

Q13) Use $nin to find exclude products with stock 10, 15 units.

-> db.Products.find({Stock:{$nin:[10,15]}})

-> Output:



Q14) Find products with Stock less than 5 and Price greater than 10.

-> db.Products.find({Stock:{$lt:5}, Price:{$gt:10}})

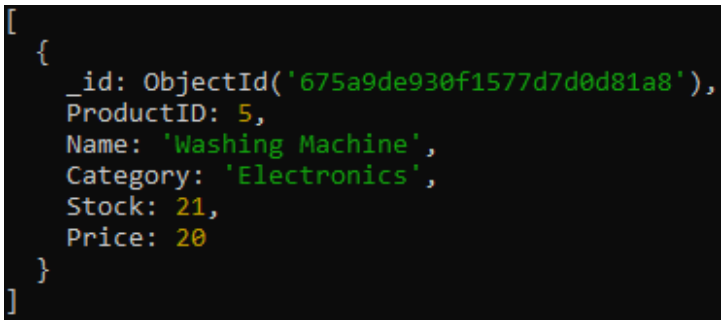
-> Output:

```
[
  {
    _id: ObjectId('675aeefa22f8a148020d8190'),
    CustomerID: 1,
    Name: 'Pratyush',
    AccountBalance: 7000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8193'),
    CustomerID: 4,
    Name: 'Suchismita',
    AccountBalance: 6000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8194'),
    CustomerID: 5,
    Name: 'Deep',
    AccountBalance: 1500,
    AccountType: 'Checking'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8195'),
    CustomerID: 6,
    Name: 'Pankaj',
    AccountBalance: 3000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8196'),
    CustomerID: 7,
    Name: 'Praveen',
    AccountBalance: 9000,
    AccountType: 'Savings'
  },
```

Q15) Find products with a Name containing the character 'c'.

-> db.Products.find({Name:{$regex:/c/}})

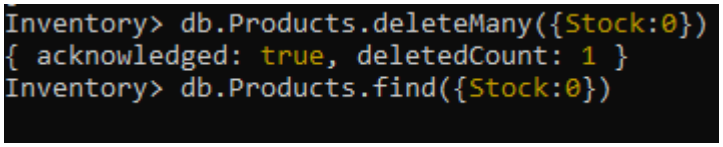
-> Output:



Q16) Delete all products with stock equal to 0.

-> db.Products.deleteMany({Stock:0})

-> Output:



Q17) Retrieve the first 5 products, skipping the first 2 documents.

-> db.Products.find().skip(2).limit(5)

-> Output:

```
[
  {
    _id: ObjectId('675a9de930f1577d7d0d81a8'),
    ProductID: 5,
    Name: 'Washing Machine',
    Category: 'Electronics',
    Stock: 21,
    Price: 20
  },
  {
    _id: ObjectId('675a9de930f1577d7d0d81a9'),
    ProductID: 6,
    Name: 'Door',
    Category: 'Furniture',
    Stock: 8,
    Price: 25
  },
  {
    _id: ObjectId('675a9de930f1577d7d0d81aa'),
    ProductID: 7,
    Name: 'Sofa',
    Category: 'Furniture',
    Stock: 7,
    Price: 9
  },
  {
    _id: ObjectId('675a9de930f1577d7d0d81ab'),
    ProductID: 8,
    Name: 'Jeans',
    Category: 'Clothing',
    Stock: 15,
    Price: 16
  },
  {
    _id: ObjectId('675a9de930f1577d7d0d81ac'),
    ProductID: 9,
    Name: 'Shoes',
    Category: 'Footwear',
    Stock: 5,
    Price: 13
  }
]
```

B) Banking System

Q1) Create a database called Bank.

-> use Bank

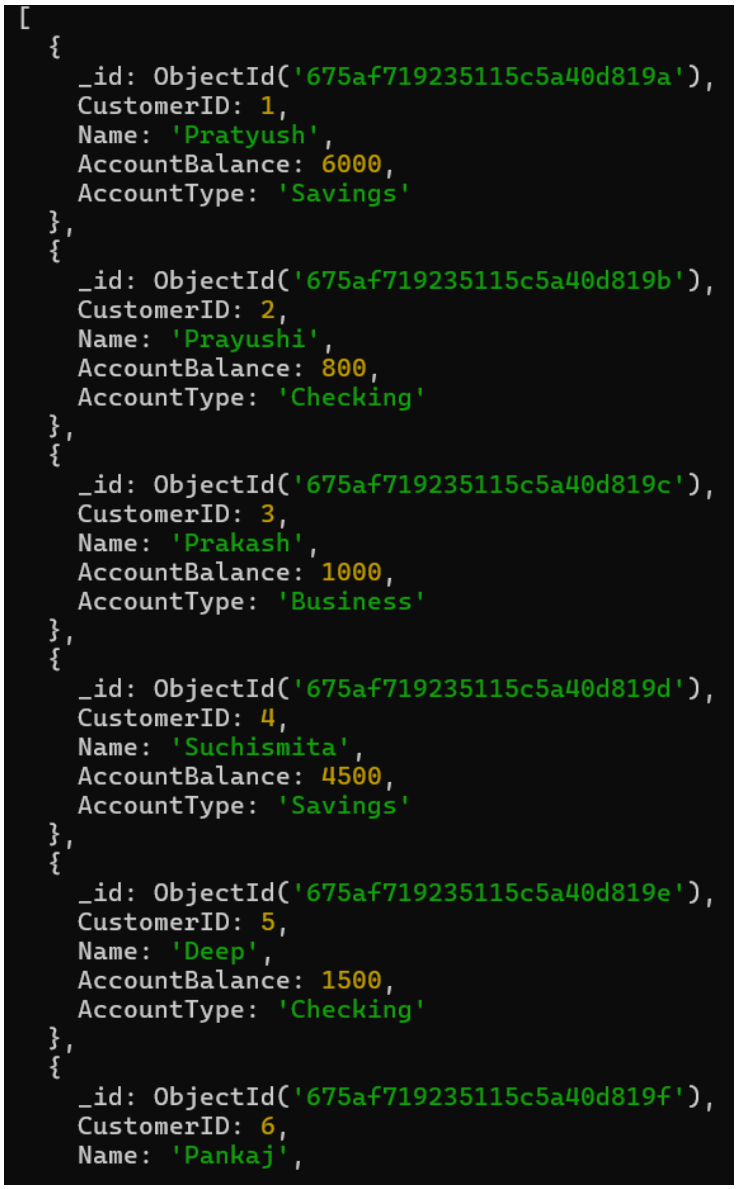
Q2) Create a collection called Customers with attributes: CustomerID, Name, AccountBalance, and AccountType.

-> db.createCollection("Customers")

Q3) Insert 10 documents representing customer information.

-> db.Customers.insertMany([{CustomerID: 1, Name: "Pratyush", AccountBalance: 6000, AccountType: "Savings" }, { CustomerID: 2, Name: "Prayushi", AccountBalance: 800, AccountType: "Checking" }, { CustomerID: 3, Name: "Prakash", AccountBalance: 1000, AccountType: "Business" }, { CustomerID: 4, Name: "Suchismita", AccountBalance: 4500, AccountType: "Savings"}, { CustomerID: 5, Name: "Deep", AccountBalance: 1500, AccountType: "Checking" }, { CustomerID: 6, Name: "Pankaj", AccountBalance: 2000, AccountType: "Savings"}, { CustomerID: 7, Name: "Praveen", AccountBalance: 8000, AccountType: "Savings"}, { CustomerID: 8, Name: "Pranab", AccountBalance: 3500, AccountType: "Business" }, { CustomerID: 9, Name: "Eshana", AccountBalance: 500, AccountType: "Savings"}, { CustomerID: 10, Name: "Shubhashish", AccountBalance: 300, AccountType: "Savings"}])

-> Output:

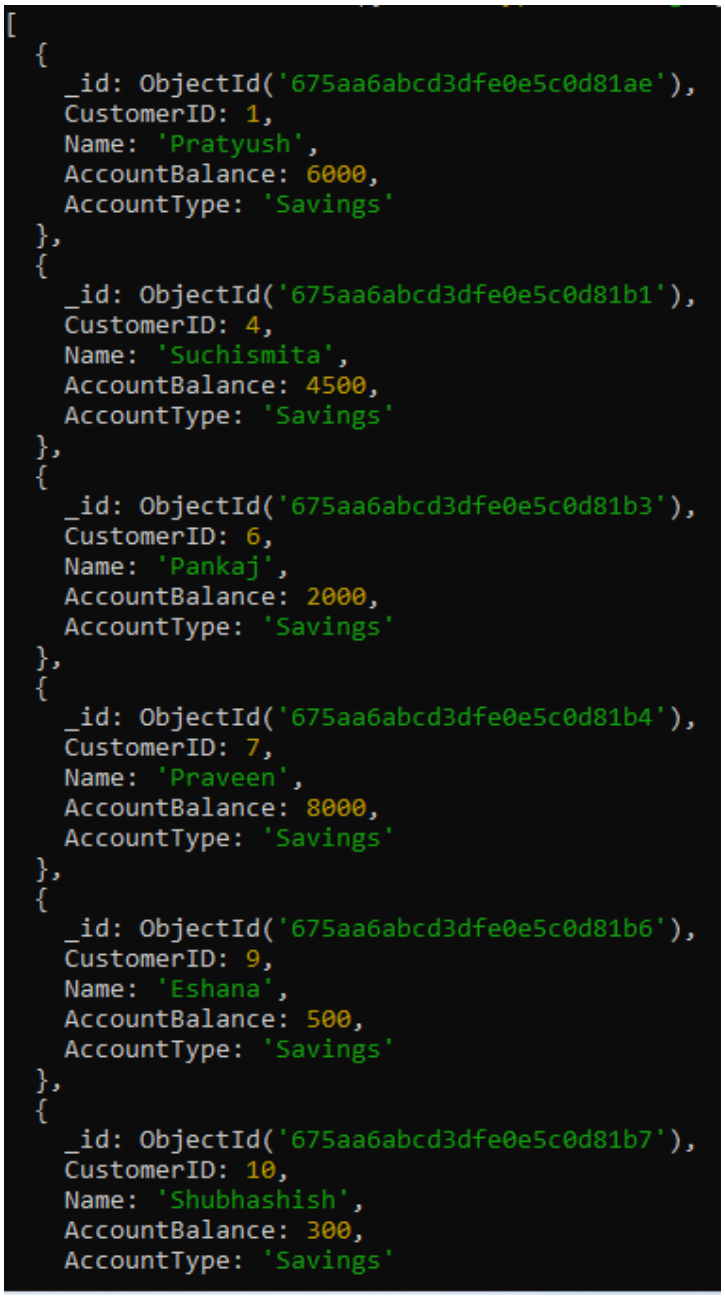


```
[
  {
    _id: ObjectId('675af719235115c5a40d819a'),
    CustomerID: 1,
    Name: 'Pratyush',
    AccountBalance: 6000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675af719235115c5a40d819b'),
    CustomerID: 2,
    Name: 'Prayushi',
    AccountBalance: 800,
    AccountType: 'Checking'
  },
  {
    _id: ObjectId('675af719235115c5a40d819c'),
    CustomerID: 3,
    Name: 'Prakash',
    AccountBalance: 1000,
    AccountType: 'Business'
  },
  {
    _id: ObjectId('675af719235115c5a40d819d'),
    CustomerID: 4,
    Name: 'Suchismita',
    AccountBalance: 4500,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675af719235115c5a40d819e'),
    CustomerID: 5,
    Name: 'Deep',
    AccountBalance: 1500,
    AccountType: 'Checking'
  },
  {
    _id: ObjectId('675af719235115c5a40d819f'),
    CustomerID: 6,
    Name: 'Pankaj',
```

Q4) Find all customers with AccountType 'Savings'.

-> db.Customers.find({AccountType: "Savings"})

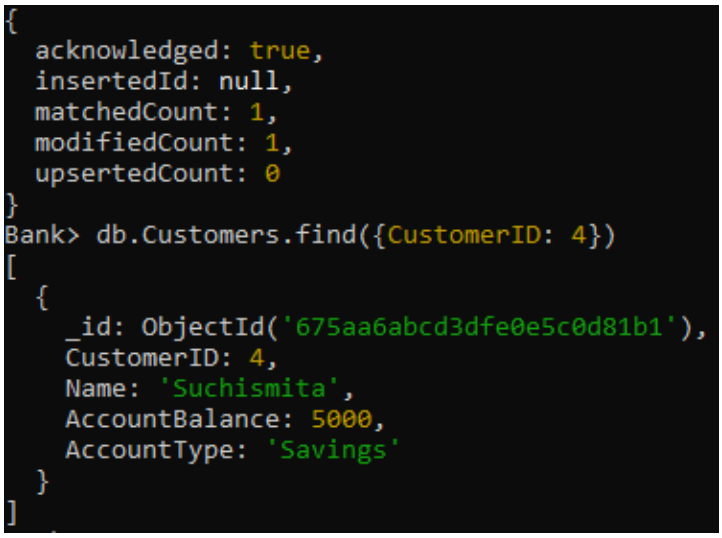
-> Output:



Q5) Update the AccountBalance of the customer with CustomerID 4 to 5000.

-> db.Customers.updateOne({CustomerID: 4}, {$set:{AccountBalance: 5000}})

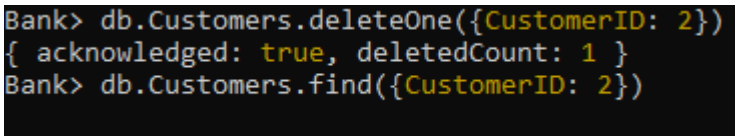
-> Output:



Q6) Delete the customer with CustomerID 2.

-> db.Customers.deleteOne({CustomerID: 2})

-> Output:



Q7) Count the number of customers with an AccountBalance greater than 1000.

-> db.Customers.find({AccountBalance:{$gt: 1000}}).count()

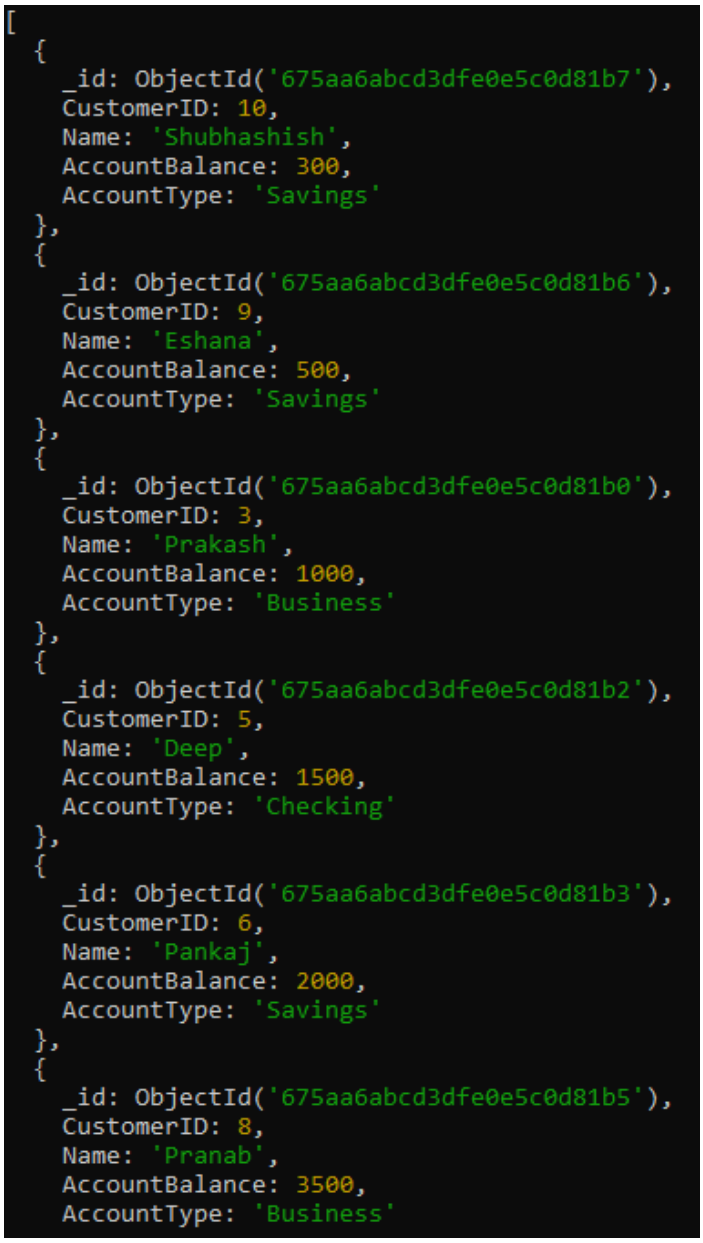
-> Output:

```
Bank> db.Customers.find({AccountBalance:{$gt: 1000}}).count()
6
```

Q8) Sort customers by AccountBalance in ascending order.

-> db.Customers.find().sort({AccountBalance:1})

-> Output:



```
[
  {
    _id: ObjectId('675aa6abcd3dfe0e5c0d81b7'),
    CustomerID: 10,
    Name: 'Shubhashish',
    AccountBalance: 300,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aa6abcd3dfe0e5c0d81b6'),
    CustomerID: 9,
    Name: 'Eshana',
    AccountBalance: 500,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aa6abcd3dfe0e5c0d81b0'),
    CustomerID: 3,
    Name: 'Prakash',
    AccountBalance: 1000,
    AccountType: 'Business'
  },
  {
    _id: ObjectId('675aa6abcd3dfe0e5c0d81b2'),
    CustomerID: 5,
    Name: 'Deep',
    AccountBalance: 1500,
    AccountType: 'Checking'
  },
  {
    _id: ObjectId('675aa6abcd3dfe0e5c0d81b3'),
    CustomerID: 6,
    Name: 'Pankaj',
    AccountBalance: 2000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aa6abcd3dfe0e5c0d81b5'),
    CustomerID: 8,
    Name: 'Pranab',
    AccountBalance: 3500,
    AccountType: 'Business'
```

Q9) Find customers whose AccountType is either 'Checking' or 'Business'.

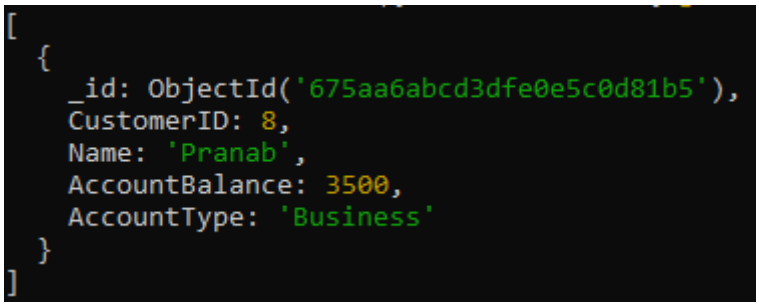
-> db.Customers.find({AccountType:{$in:["Checking", "Business"]}})

-> Output:

Q10) Find customers with an AccountBalance between 2000 and 5000.

-> db.Customers.find({AccountBalance:{$gt:2000, $lt:5000}})

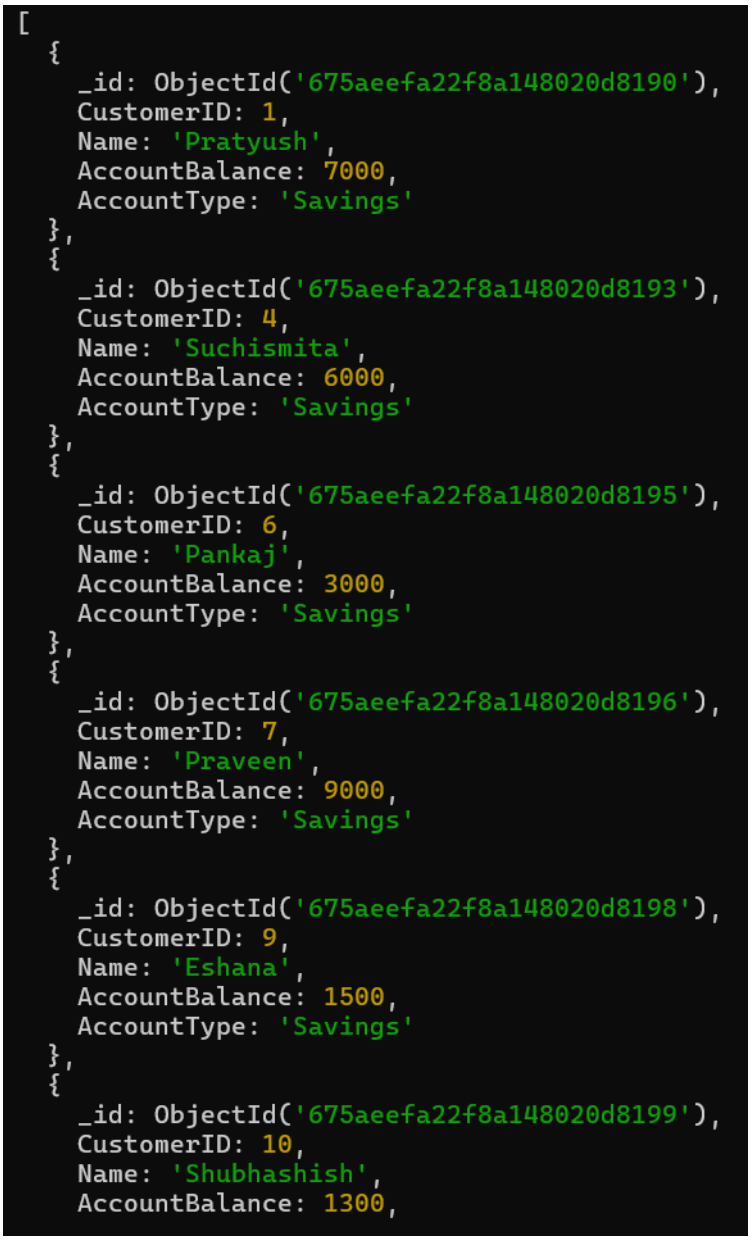
-> Output:



Q11) Update the AccountBalance of all customers with AccountType 'Savings' by 1000.

-> db.Customers.updateMany({AccountType: "Savings"}, {$inc:{AccountBalance: 1000}})

-> Output:



```
[
  {
    _id: ObjectId('675aeefa22f8a148020d8190'),
    CustomerID: 1,
    Name: 'Pratyush',
    AccountBalance: 7000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8193'),
    CustomerID: 4,
    Name: 'Suchismita',
    AccountBalance: 6000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8195'),
    CustomerID: 6,
    Name: 'Pankaj',
    AccountBalance: 3000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8196'),
    CustomerID: 7,
    Name: 'Praveen',
    AccountBalance: 9000,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8198'),
    CustomerID: 9,
    Name: 'Eshana',
    AccountBalance: 1500,
    AccountType: 'Savings'
  },
  {
    _id: ObjectId('675aeefa22f8a148020d8199'),
    CustomerID: 10,
    Name: 'Shubhashish',
    AccountBalance: 1300,
```

Q12) Retrieve customers with an AccountBalance greater than 5000.

-> db.Customers.find({AccountBalance:{$gt:5000}})

-> Output:

Q13) Find customers with AccountType 'Checking' and AccountBalance less than 1000.

-> db.Customers.find({AccountType: "Checking", AccountBalance:{$lt:1000}})

-> Output:

```
Bank> db.Customers.find({AccountType: "Checking", AccountBalance:{$lt:1000}})

Bank> db.Customers.find({AccountType: "Checking"})
[
  {
    _id: ObjectId('675aa6abcd3dfe0e5c0d81b2'),
    CustomerID: 5,
    Name: 'Deep',
    AccountBalance: 1500,
    AccountType: 'Checking'
  }
]
```

Q14) Use $in to find customers with AccountType either 'Business' or 'Checking'.

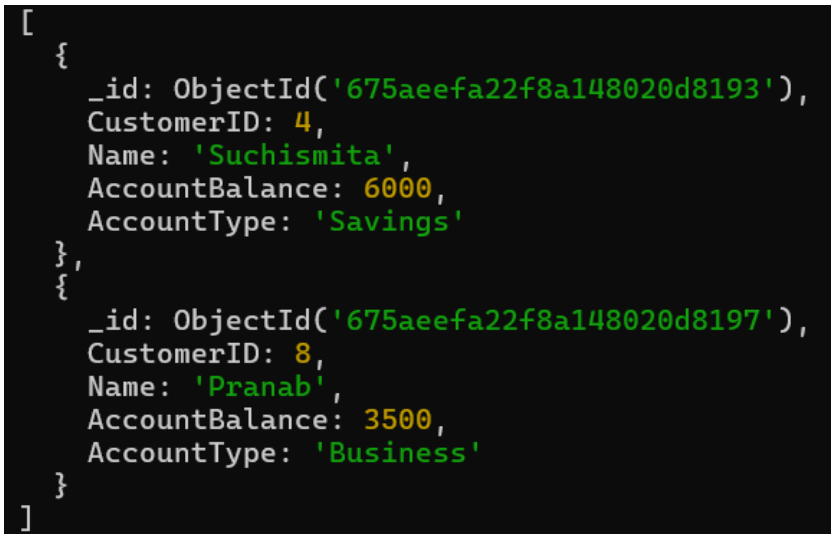
-> db.Customers.find({AccountType:{$in:["Checking", "Savings"]}})

-> Output:

Q15) Find customers whose AccountBalance is between 3000 and 7000.

-> db.Customers.find({AccountBalance:{$gt:3000, $lt:7000}})

-> Output:



Q16) Find all customers with AccountBalance less than 1000 and update it to 1500.

-> db.Customers.updateMany({AccountBalance:{$lt:1000}}, {$set:{AccountBalance: 1500}})

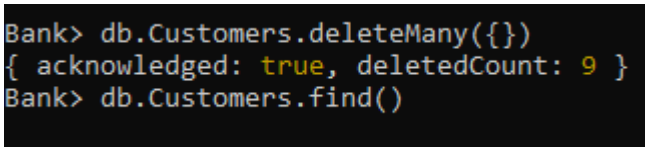
-> Output:

```
Bank> db.Customers.updateMany({AccountBalance:{$lt:1000}}, {$set:{AccountBalance: 1500}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
Bank> db.Customers.find({AccountBalance:{$lt:1000}})
```

Q17) Delete all the customers.

-> db.Customers.deleteMany({})

-> Output:



C) Book and Author Database

Q1) Create a database called "Library".

-> use Library

Q2) Create a collection called "Books" with attributes: BookID, Title, AuthorID, Genre, PublishedYear, and Pages.

-> db.createCollection("Books")

Q3) Create a collection called "Authors" with attributes: AuthorID, Name, DateOfBirth, Nationality.

-> db.createCollection("Authors")

Q4) Insert 5 books into the "Books" collection.

-> db.Books.insertMany([{BookID: 1, Title: "Harry Potter", AuthorID: 123, Genre: "Fiction", PublishedYear: 2004, Pages: 100}, {BookID: 2, Title: "Poseidon", AuthorID: 124, Genre: "Science Fiction", PublishedYear: 1998, Pages: 80}, {BookID: 3, Title: "Think and Grow Rich", AuthorID: 125, Genre: "Thriller", PublishedYear: 2008, Pages: 88}, {BookID: 4, Title: "Atomic Habits", AuthorID: 123, Genre: "Romance", PublishedYear: 2001, Pages: 150}, {BookID: 5, Title: "Rich Dad Poor Dad", AuthorID: 124, Genre: "Adventure", PublishedYear: 1990, Pages: 60}, {BookID: 6, Title: "Ikigai", AuthorID: 125, Genre: "Horror", PublishedYear: 2003, Pages: 90}])

-> Output:


```
[
  {
    _id: ObjectId('675af760235115c5a40d81a4'),
    BookID: 1,
    Title: 'Harry Potter',
    AuthorID: 123,
    Genre: 'Fiction',
    PublishedYear: 2004,
    Pages: 100
  },
  {
    _id: ObjectId('675af760235115c5a40d81a5'),
    BookID: 2,
    Title: 'Poseidon',
    AuthorID: 124,
    Genre: 'Science Fiction',
    PublishedYear: 1998,
    Pages: 80
  },
  {
    _id: ObjectId('675af760235115c5a40d81a6'),
    BookID: 3,
    Title: 'Think and Grow Rich',
    AuthorID: 125,
    Genre: 'Thriller',
    PublishedYear: 2008,
    Pages: 88
  },
  {
    _id: ObjectId('675af760235115c5a40d81a7'),
    BookID: 4,
    Title: 'Atomic Habits',
    AuthorID: 123,
    Genre: 'Romance',
    PublishedYear: 2001,
    Pages: 150
  },
  {
    _id: ObjectId('675af760235115c5a40d81a8'),
    BookID: 5,
    Title: 'Rich Dad Poor Dad',
```

Q5) Insert 3 authors into the "Authors" collection.

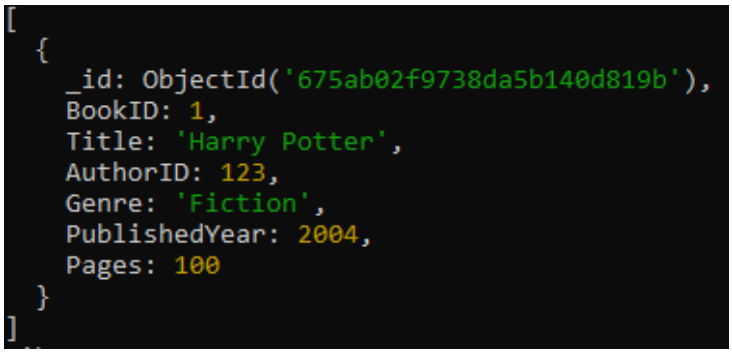
-> db.Authors.insertMany([{AuthorID: 123, Name: "J.K. Rowling", DateOfBirth: 1982, Nationality: "American"}, {AuthorID: 124, Name: "Agatha Christy", DateOfBirth: 1970, Nationality: "Turkish"}, {AuthorID: 125, Name: "Ruskin Bond", DateOfBirth: 1995, Nationality: "Spanish"}])

-> Output:

Q6) Find all books with the Genre "Fiction".

-> db.Books.find({Genre: "Fiction"})

-> Output:



Q7) Find all books authored by an author with the Name "J.K. Rowling".

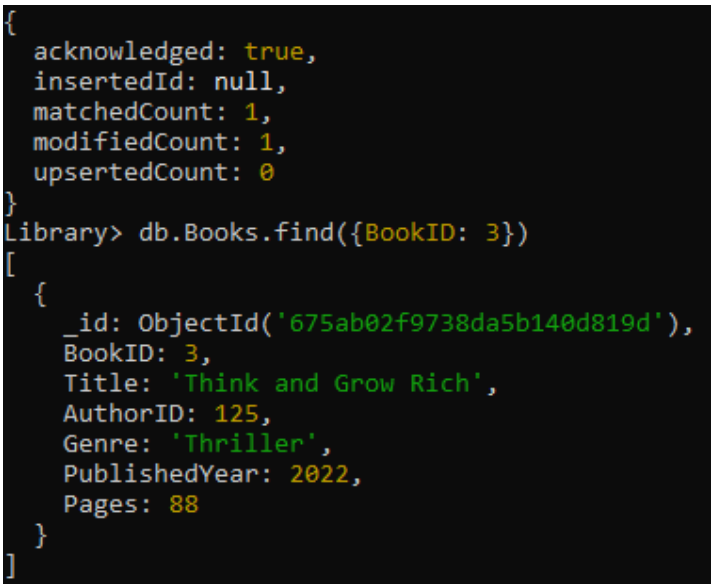
-> const author = db.Authors.findOne({Name: "J.K. Rowling"})
db.Books.find({AuthorID: author.AuthorID})

-> Output:

Q8) Update the PublishedYear of the book with BookID 3 to 2022.

-> db.Books.updateOne({BookID: 3}, {$set:{PublishedYear: 2022}})

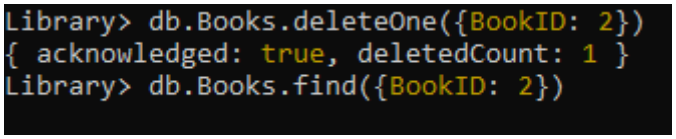
-> Output:



Q9) Delete the book with BookID 2.

-> db.Books.deleteOne({BookID: 2})

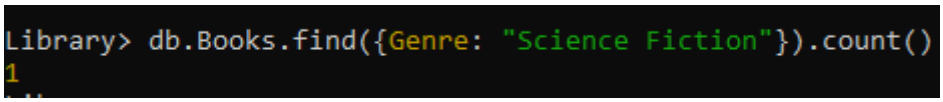
-> Output:



Q10) Count how many books belong to the "Science Fiction" genre.

-> db.Books.find({Genre: "Science Fiction"}).count()

-> Output:



Q11) Find books published between the years 2000 and 2010.

-> db.Books.find({PublishedYear:{$gt:2000, $lt:2010}})

-> Output:

```
[
  {
    _id: ObjectId('675ab02f9738da5b140d819b'),
    BookID: 1,
    Title: 'Harry Potter',
    AuthorID: 123,
    Genre: 'Fiction',
    PublishedYear: 2004,
    Pages: 100
  },
  {
    _id: ObjectId('675ab02f9738da5b140d819e'),
    BookID: 4,
    Title: 'Atomic Habits',
    AuthorID: 123,
    Genre: 'Romance',
    PublishedYear: 2001,
    Pages: 150
  },
  {
    _id: ObjectId('675ab02f9738da5b140d81a0'),
    BookID: 6,
    Title: 'Ikigai',
    AuthorID: 125,
    Genre: 'Horror',
    PublishedYear: 2003,
    Pages: 90
  }
]
```

Q12) Use $in to find books that belong to the genres "Thriller", "Romance", or "Adventure".

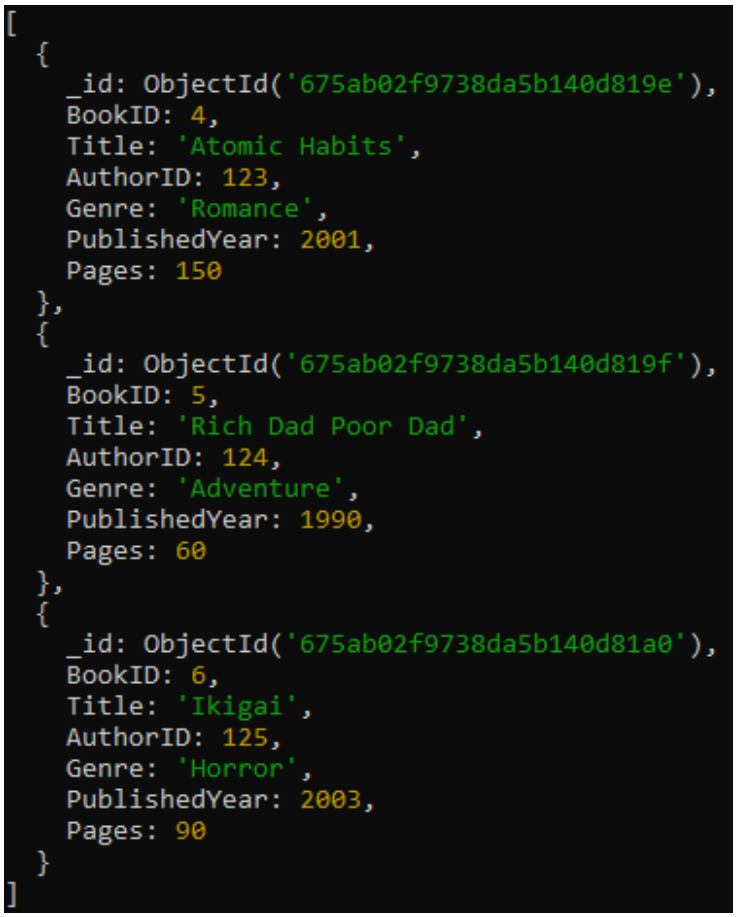
-> db.Books.find({Genre:{$in:["Thriller", "Romance", "Adventure"]}})

-> Output:

Q13) Retrieve the first 3 books, skipping the first 2 documents.

-> db.Books.find().skip(2).limit(3)

-> Output:

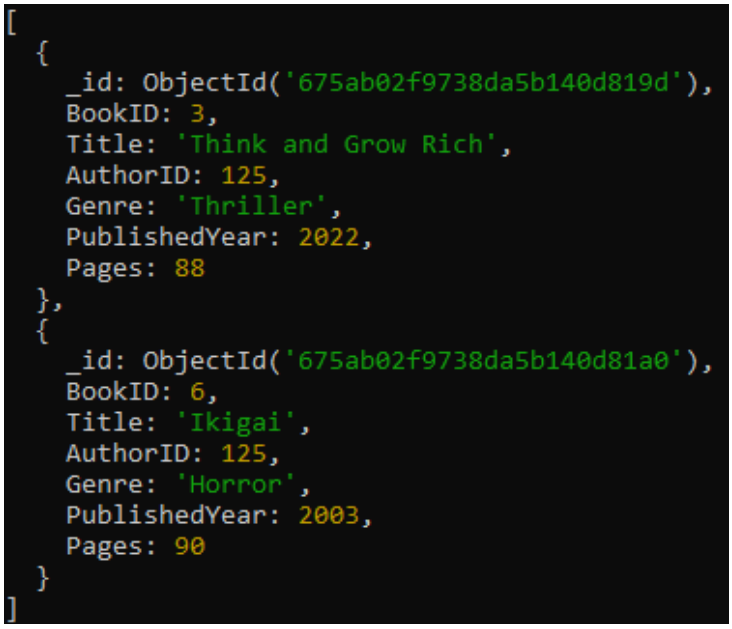


```
[
  {
    _id: ObjectId('675ab02f9738da5b140d819e'),
    BookID: 4,
    Title: 'Atomic Habits',
    AuthorID: 123,
    Genre: 'Romance',
    PublishedYear: 2001,
    Pages: 150
  },
  {
    _id: ObjectId('675ab02f9738da5b140d819f'),
    BookID: 5,
    Title: 'Rich Dad Poor Dad',
    AuthorID: 124,
    Genre: 'Adventure',
    PublishedYear: 1990,
    Pages: 60
  },
  {
    _id: ObjectId('675ab02f9738da5b140d81a0'),
    BookID: 6,
    Title: 'Ikigai',
    AuthorID: 125,
    Genre: 'Horror',
    PublishedYear: 2003,
    Pages: 90
  }
]
```

Q14) Find all books written by authors born after 1980.

-> const author_name = db.Authors.findOne({DateOfBirth:{$gt:1980}, Name: "Ruskin Bond"})
db.Books.find({AuthorID: author_name.AuthorID})

-> Output:



Q15) Sort books by Title in alphabetical order.

-> db.Books.find().sort({Title: 1})

-> Output:

```
[
  {
    _id: ObjectId('675ab02f9738da5b140d819e'),
    BookID: 4,
    Title: 'Atomic Habits',
    AuthorID: 123,
    Genre: 'Romance',
    PublishedYear: 2001,
    Pages: 150
  },
  {
    _id: ObjectId('675ab02f9738da5b140d819b'),
    BookID: 1,
    Title: 'Harry Potter',
    AuthorID: 123,
    Genre: 'Fiction',
    PublishedYear: 2004,
    Pages: 100
  },
  {
    _id: ObjectId('675ab02f9738da5b140d81a0'),
    BookID: 6,
    Title: 'Ikigai',
    AuthorID: 125,
    Genre: 'Horror',
    PublishedYear: 2003,
    Pages: 90
  },
  {
    _id: ObjectId('675ab02f9738da5b140d819f'),
    BookID: 5,
    Title: 'Rich Dad Poor Dad',
    AuthorID: 124,
    Genre: 'Adventure',
    PublishedYear: 1990,
    Pages: 60
  },
  {
    _id: ObjectId('675ab02f9738da5b140d819d'),
    BookID: 3,
    Title: 'Think and Grow Rich',
    AuthorID: 125,
```

Q16) Group books by Genre and find the average PublishedYear for each genre.

-> db.Books.aggregate([{$group:{_id:"$Genre", avgPublicationYear:{$avg: "$PublishedYear"}}}])

-> Output:

```
Library> db.Books.aggregate([{$group:{_id:"$Genre", avgPublicationYear:{$avg: "$PublishedYear"}}}])
[
  { _id: 'Romance', avgPublicationYear: 2001 },
  { _id: 'Thriller', avgPublicationYear: 2022 },
  { _id: 'Adventure', avgPublicationYear: 1990 },
  { _id: 'Fiction', avgPublicationYear: 2004 },
  { _id: 'Horror', avgPublicationYear: 2003 }
]
```

Q17) Find all books whose Title contains the word "Harry Potter".

-> db.Books.find({Title: "Harry Potter"})

-> Output:

```
Library> db.Books.find({Title: "Harry Potter"})
[
  {
    _id: ObjectId('675ab02f9738da5b140d819b'),
    BookID: 1,
    Title: 'Harry Potter',
    AuthorID: 123,
    Genre: 'Fiction',
    PublishedYear: 2004,
    Pages: 100
  }
]
```

Q18) Find all authors who have written books in the "Horror" genre.

-> const book = db.Books.findOne({Genre: "Horror"})
db.Authors.find({AuthorID: book.AuthorID})

-> Output: