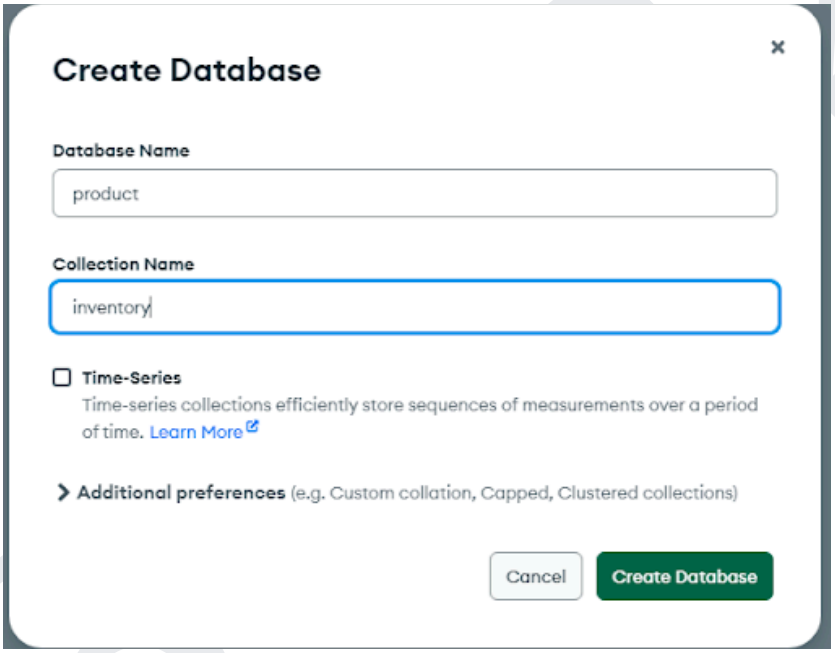
Name : Avdhut Pailwan
PRN : 2122000380
Roll No. : BTB22
Subject : Advanced Database Systems Lab

Experiment No. : 7

Install MongoDB Compass. Create and manage NoSQL Databases with MongoDB

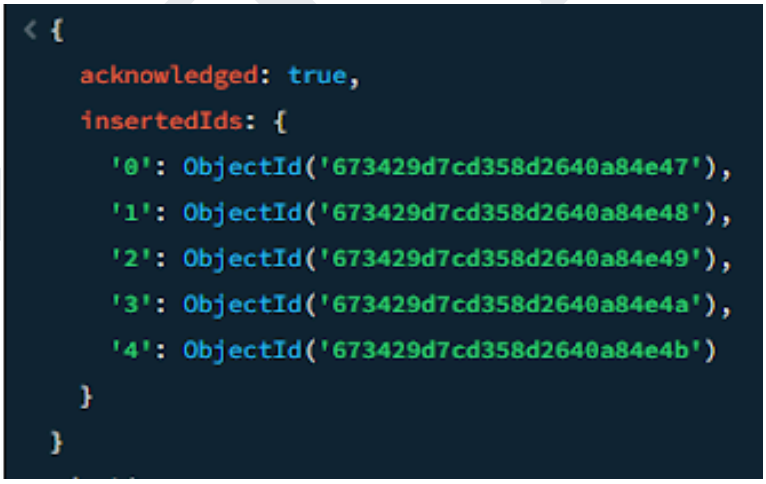
Problem Statement 1:

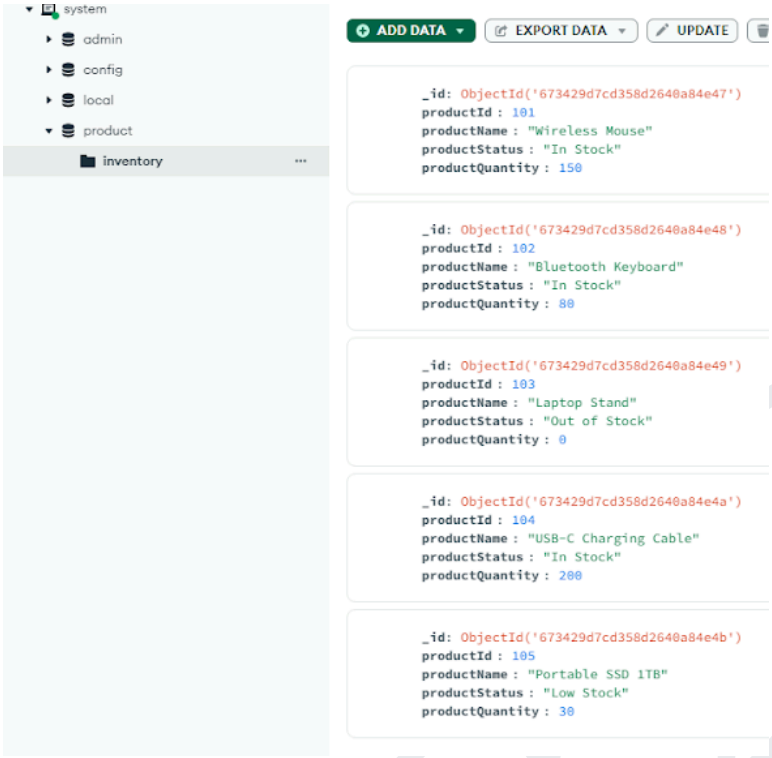
1. Create database: product
2. Create collection: inventory



3. Perform following operations on created collections:

a. Insert documents (one and many)






```
product> db["inventory"].insertOne({
            productId: 106,
            productName: "Noise Cancelling Headphones",
            productStatus: "In Stock",
            productQuantity: 75
          });
```

```
{
  acknowledged: true,
  insertedId: ObjectId('67342cdd64f2f54333a0e804')
}
```

```
productStatus : "Low Stock"
productQuantity : 30
```

```
_id: ObjectId('67342cdd64f2f54333a0e804')
productId : 106
productName : "Noise Cancelling Headphones"
productStatus : "In Stock"
productQuantity : 75
```

b. Update documents (one and many).

```
product> db["inventory"].updateOne(
          { productId: 101 },
          { $set: { productQuantity: 180 } }
        );
```

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
_id: ObjectId('673429d7cd358d2640a84e47')
productId : 101
productName : "Wireless Mouse"
productStatus : "In Stock"
productQuantity : 180
```

```
_id: ObjectId('673429d7cd358d2640a84e47')
productId : 101
productName : "Wireless Mouse"
productStatus : "A"
productQuantity : 180
```

```
_id: ObjectId('673429d7cd358d2640a84e49')
productId : 103
productName : "Laptop Stand"
productStatus : "A"
productQuantity : 0
```

```
_id: ObjectId('67342cdd64f2f54333a0e804')
productId : 106
productName : "Noise Cancelling Headphones"
productStatus : "A"
productQuantity : 75
```

```
product> db["inventory"].updateMany(
          {"productStatus": "A"},
          {$set: {"productQuantity": 100}}
        )
```

```
_id: ObjectId('673429d7cd358d2640a84e47')
productId : 101
productName : "Wireless Mouse"
productStatus : "A"
productQuantity : 100
```

```
_id: ObjectId('673429d7cd358d2640a84e49')
productId : 103
productName : "Laptop Stand"
productStatus : "A"
productQuantity : 100
```

```
_id: ObjectId('67342cdd64f2f54333a0e804')
productId : 106
productName : "Noise Cancelling Headphones"
productStatus : "A"
productQuantity : 100
```

c. Replace documents (one and many).

```
_id: ObjectId('673429d7cd358d2640a84e47')
productId : 101
productName : "Wireless Mouse"
productStatus : "A"
productQuantity : 100
```

```
product> db["inventory"].replaceOne(
          {"productId": 101},
          {
            productId: 101,
            productName: "Ergonomic Wireless Mouse",
            productStatus: "D",
            productQuantity: 120
          }
        );
```

```
_id: ObjectId('673429d7cd358d2640a84e47')
productId : 101
productName : "Ergonomic Wireless Mouse"
productStatus : "D"
productQuantity : 120
```

d. Delete documents (one and many).

```
product> db["inventory"].deleteOne(
            {"productId": 106}
          );
```

```
{
  acknowledged: true,
  deletedCount: 1
}
```

```
product> db["inventory"].deleteMany(
           {"productStatus": "A"}
         );
```

```
{
  acknowledged: true,
  deletedCount: 3
}
```

e. Find documents.

4. Use a filter to find documents in the database. Perform following queries in filter on inventory collection.

a. `SELECT * FROM inventory`

```
> db["inventory"].find();
< {
    _id: ObjectId('673429d7cd358d2640a84e47'),
    productId: 101,
    productName: 'Ergonomic Wireless Mouse',
    productStatus: 'D',
    productQuantity: 120
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e48'),
    productId: 102,
    productName: 'Bluetooth Keyboard',
    productStatus: 'D',
    productQuantity: 80
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e4a'),
    productId: 104,
    productName: 'USB-C Charging Cable',
    productStatus: 'A',
    productQuantity: 200
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e4b'),
    productId: 105,
    productName: 'Portable SSD 1TB',
    productStatus: 'D',
    productQuantity: 30
  }
```

b. `SELECT * FROM inventory WHERE status = "D"`

```
> db["inventory"].find({"productStatus": "D"});
< {
    _id: ObjectId('673429d7cd358d2640a84e47'),
    productId: 101,
    productName: 'Ergonomic Wireless Mouse',
    productStatus: 'D',
    productQuantity: 120
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e48'),
    productId: 102,
    productName: 'Bluetooth Keyboard',
    productStatus: 'D',
    productQuantity: 80
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e4b'),
    productId: 105,
    productName: 'Portable SSD 1TB',
    productStatus: 'D',
    productQuantity: 30
  }
```

c. `SELECT * FROM inventory WHERE status in ("A", "D")`

```
> db["inventory"].find(
    {"productStatus": {$in: ["A", "D"]}}
  );
< {
    _id: ObjectId('673429d7cd358d2640a84e47'),
    productId: 101,
    productName: 'Ergonomic Wireless Mouse',
    productStatus: 'D',
    productQuantity: 120
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e48'),
    productId: 102,
    productName: 'Bluetooth Keyboard',
    productStatus: 'D',
    productQuantity: 80
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e4a'),
    productId: 104,
    productName: 'USB-C Charging Cable',
    productStatus: 'A',
    productQuantity: 200
  }
  {
    _id: ObjectId('673429d7cd358d2640a84e4b'),
    productId: 105,
    productName: 'Portable SSD 1TB',
    productStatus: 'D',
    productQuantity: 30
  }
```

d. `SELECT * FROM inventory WHERE status = "A" AND qty < 30`

```
> db["inventory"].find(
    {
      "productStatus": "A",
      "productQuantity": {$lt: 30}
    }
  )
<
```

e. `SELECT * FROM inventory WHERE status = "A" OR qty < 30`

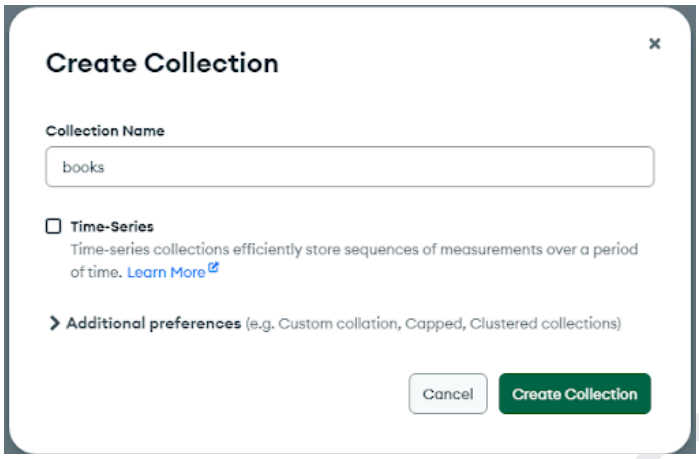
```
> db["inventory"].find(
    {
      $or: [
        {"productStatus": "A"},
        {"productQuantity": {$lt: 30}}
      ]
    }
  );
< {
    _id: ObjectId('673429d7cd358d2640a84e4a'),
    productId: 104,
    productName: 'USB-C Charging Cable',
    productStatus: 'A',
    productQuantity: 200
  }
```

f. `SELECT * FROM inventory WHERE status = "A" AND ( qty < 30 OR item LIKE "p%")`

```
> db["inventory"].find(
    {
      status: "A",
      $or: [
        { qty: { $lt: 30 } },
        { item: { $regex: "^p", $options: "i" } }
      ]
    }
  );
<
```

Problem Statement 2:

- Create collection: books under product database



- Insert the following documents into a books collection:

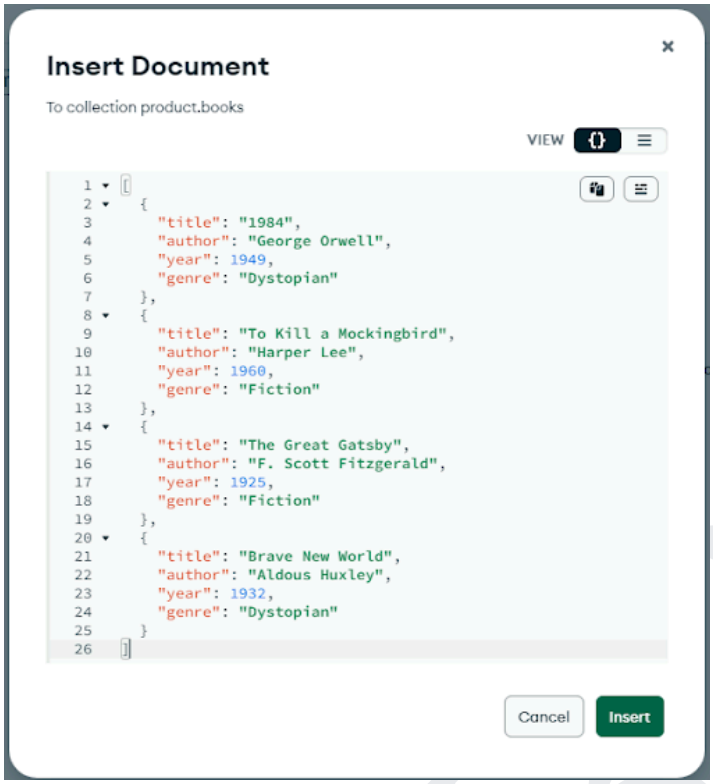
  { "title": "1984", "author": "George Orwell", "year": 1949, "genre": "Dystopian" }

  { "title": "To Kill a Mockingbird", "author": "Harper Lee", "year": 1960, "genre": "Fiction" }

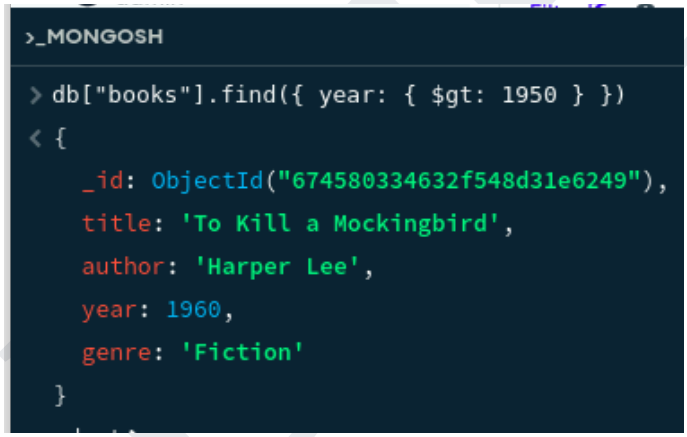
  { "title": "The Great Gatsby", "author": "F. Scott Fitzgerald", "year": 1925, "genre": "Fiction" }

  { "title": "Brave New World", "author": "Aldous Huxley", "year": 1932, "genre": "Dystopian" }

  Add more such documents.



- Find all books published after the year 1950.



- Find all Dystopian books published before 1950.

```
>_MONGOSH
> db["books"].find({ genre: "Dystopian", year: { $lt: 1950 } })
< {
    _id: ObjectId("674580334632f548d31e6248"),
    title: '1984',
    author: 'George Orwell',
    year: 1949,
    genre: 'Dystopian'
  }
  {
    _id: ObjectId("674580334632f548d31e624b"),
    title: 'Brave New World',
    author: 'Aldous Huxley',
    year: 1932,
    genre: 'Dystopian'
  }
```

- Update the genre of "1984" to "Science Fiction".\

```
>_MONGOSH
> db["books"].updateOne({ title: "1984" }, { $set: { genre: "Science Fiction" } })
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

- Delete all books in the "Fiction" genre.

```
>_MONGOSH
> db["books"].deleteMany({ genre: "Fiction" })
< {
    acknowledged: true,
    deletedCount: 2
  }
```

- Calculate the total number of books for each genre.

```
> db.books.aggregate([
    { $group: { _id: "$genre", count: { $sum: 1 } } }
  ])
< {
    _id: 'Dystopian',
    count: 1
  }
  {
    _id: 'Science Fiction',
    count: 1
  }
```

- Create an index on the author field to improve query performance.

```
>_MONGOSH

> db.books.createIndex({ author: 1 })
< author_1
```

- Retrieve all books sorted by year in ascending order.

```
>_MONGOSH
> db.books.find().sort({ year: 1 })
< {
    _id: ObjectId("674580334632f548d31e624b"),
    title: 'Brave New World',
    author: 'Aldous Huxley',
    year: 1932,
    genre: 'Dystopian'
  }
  {
    _id: ObjectId("674580334632f548d31e6248"),
    title: '1984',
    author: 'George Orwell',
    year: 1949,
    genre: 'Science Fiction'
  }
```

- Count the number of books written by "Harper Lee".

```
>_MONGOSH
> db.books.countDocuments({ author: "Harper Lee" })
< 0
product>
```

- Retrieve only the titles and authors of all books.

```
>_MONGOSH
> db.books.find({}, { _id: 0, title: 1, author: 1 })
< {
    title: '1984',
    author: 'George Orwell'
  }
  {
    title: 'Brave New World',
    author: 'Aldous Huxley'
  }
```

- Use filter to find documents in database. Perform following queries in filter on inventory collection.
    - Find books published between 1930 and 1960.

```
>_MONGOSH
> db.books.find({ year: { $gte: 1930, $lte: 1960 } })
< {
    _id: ObjectId("674580334632f548d31e6248"),
    title: '1984',
    author: 'George Orwell',
    year: 1949,
    genre: 'Science Fiction'
  }
  {
    _id: ObjectId("674580334632f548d31e624b"),
    title: 'Brave New World',
    author: 'Aldous Huxley',
    year: 1932,
    genre: 'Dystopian'
  }
```

- Find books with titles containing the word "The".

```
>_MONGOSH
> db.books.find({ title: { $regex: /The/, $options: "i" } })
<
```

- Find all books published before 1950 and in the Fiction genre.

```
>_MONGOSH
> db.books.find({ year: { $lt: 1950 }, genre: "Fiction" })
<
```

- Find all books not written by Aldous Huxley.

```
>_MONGOSH
> db.books.find({ author: { $ne: "Aldous Huxley" } })
< {
    _id: ObjectId("674580334632f548d31e6248"),
    title: '1984',
    author: 'George Orwell',
    year: 1949,
    genre: 'Science Fiction'
  }
```