# Atharva Pawar - Comps-A [Batch-D]

BDA - EXP - 4 : MongoDB CRUD Cmds

In [10]:
```
# pip install pymongo
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
```

In [11]:
```
db = client['GTA']
collection = db['mySampleCollection']
```

## Insert

In [12]:
```
new_student = {'_id':1, 'name':'Omkar','location':'mumbai', 'Marks': 90, 'Pass':'No'}
collection.insert_one(new_student)
```

Out[12]: <pymongo.results.InsertOneResult at 0x2574ac0e050>

## find()

In [9]:
```
all_students = collection.find()
for item in all_students:
    print(item)
```

```
{'_id': 5, 'name': 'Gtapawar', 'location': 'pune', 'Marks': 34}
{'_id': 1, 'name': 'Omkar', 'location': 'mumbai', 'Marks': 46}
```

In [7]:
```
specific_student = collection.find_one({'name': 'Omkar'})
print(specific_student)
```

```
{'_id': 1, 'name': 'Omkar', 'location': 'mumbai', 'Marks': 90}
```

## update_one() or update_many()

In [8]:
```
# Update a single document
collection.update_one({'name': 'Omkar'}, {'$set': {'Marks': 46}})
```

Out[8]: <pymongo.results.UpdateResult at 0x2574ac0df90>

In [ ]:
```
# Update multiple documents
collection.update_many({'Marks': {'$lt': 34}}, {'$set': {'status': 'pass'}})
```

In [29]:
```
all_students = collection.find()
for item in all_students:
    print(item)
```

```
{'_id': 1, 'name': 'Gtapawar', 'location': 'pune', 'Marks': 341234}
{'_id': 2, 'name': 'Gtapawar1', 'location': 'pune1', 'Marks': 342}
{'_id': 3, 'name': 'Gtapawar2', 'location': 'pune2', 'Marks': 342}
```

## Delete

In [24]:
```
collection.delete_one({'Marks': 34})
```

Out[24]: <pymongo.results.DeleteResult at 0x1afb7e79db0>

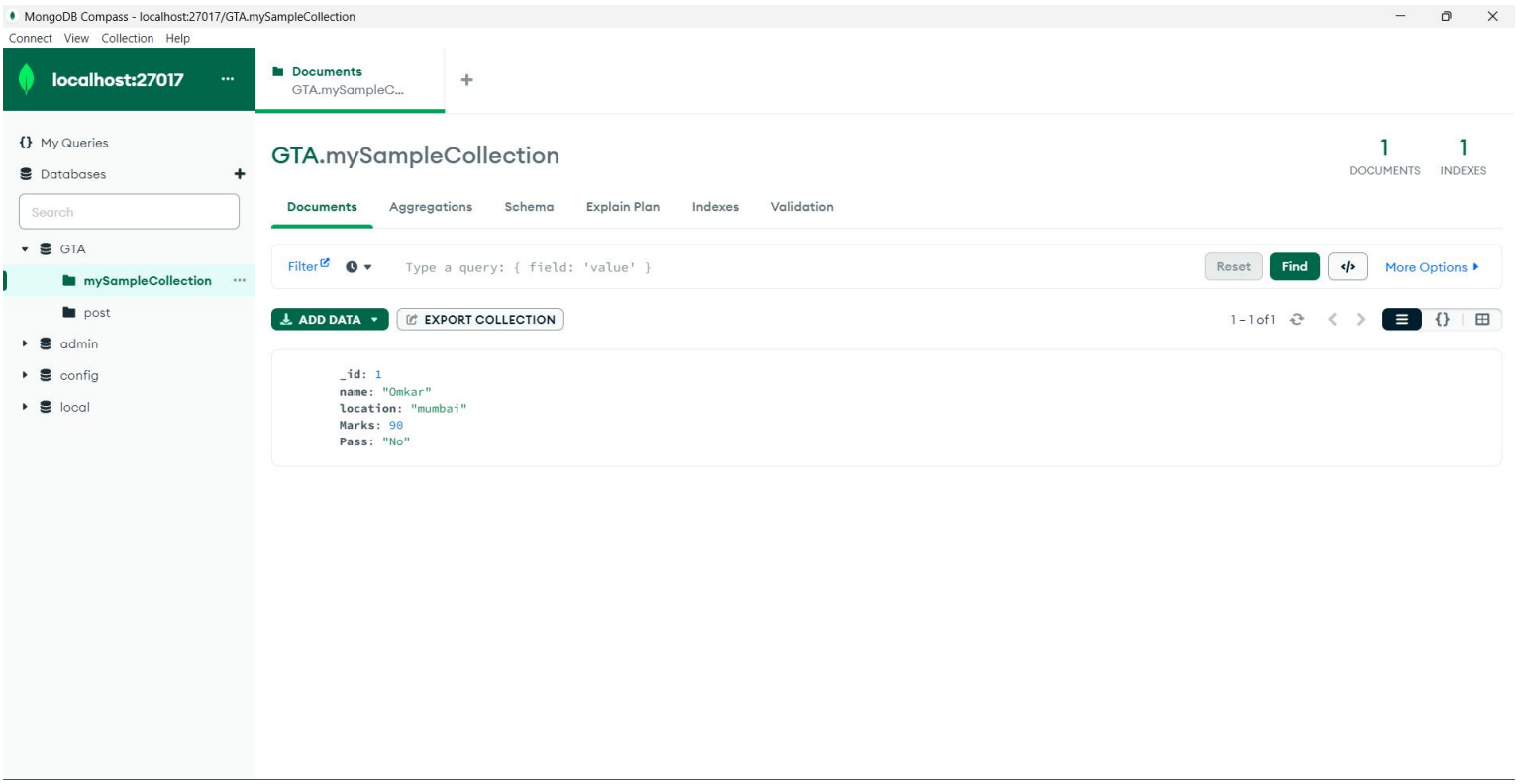
In [28]:
```
collection.delete_many({'Marks': {'$lt': 342}})
```

Out[28]: <pymongo.results.DeleteResult at 0x1afb7e79d80>

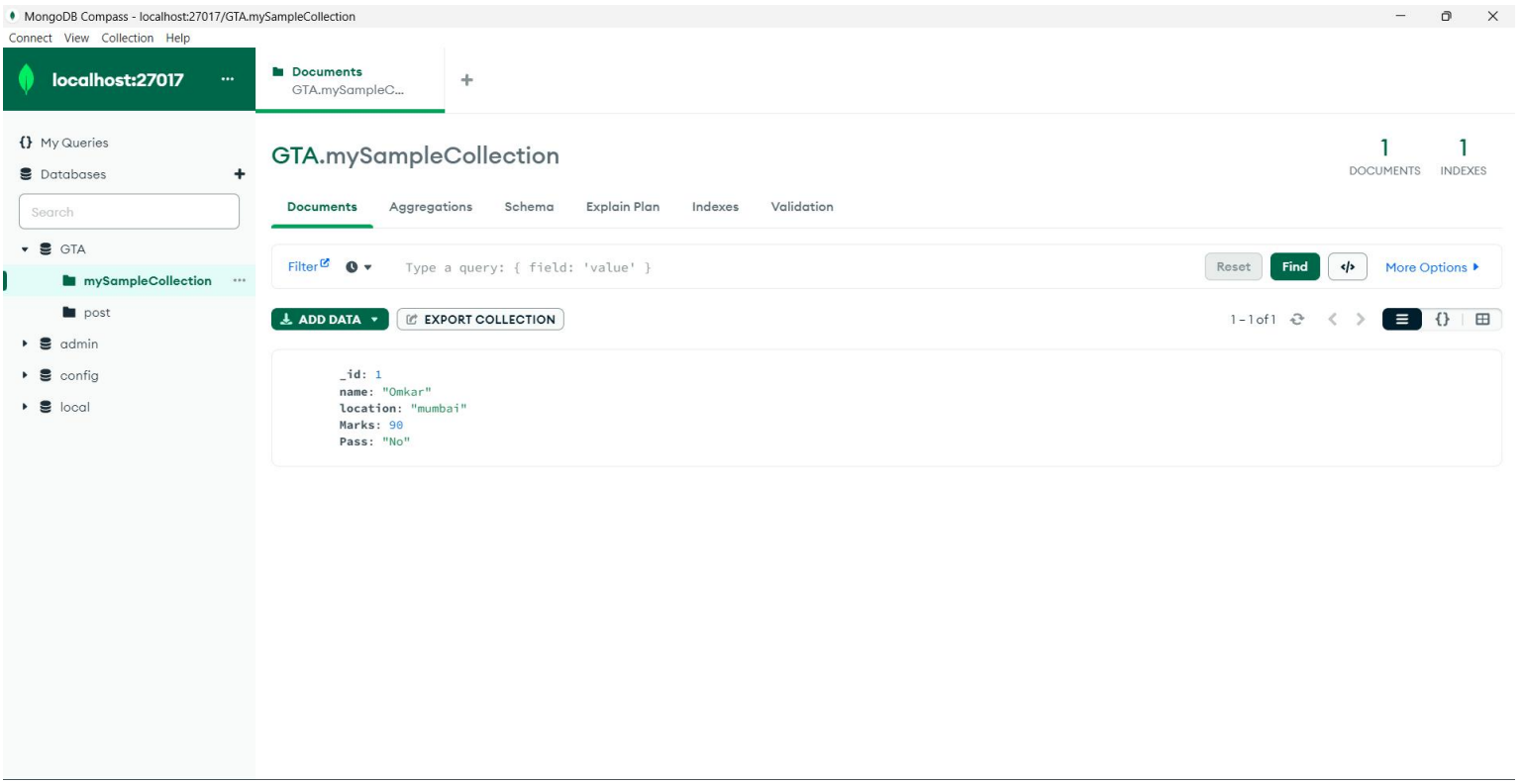
In [31]:
```
# Count docs
print(collection.count_documents({}))
```

```
3
```

# MongoDB Compass (Local)



# MongoDB Compass (Local)



Atharva Prashant Pawar (9427) -[Batch-D]

BDA Exp 4 DATE:

Q1. Define what is block in HDFS?

=>

Block in HDFS:

HDFS (Hadoop Distributed file system) is the primary storage system used in Hadoop for storing large datasets accross a cluster of machines.

HDFS divides files into smaller units known as 'blocks' to facilitate efficient storage & processing.

Blocks are the fundamental storage unit in HDFS, & their size is typically set to 128 MB or 256 MB, although it can be configured.

Unlike traditional file system that might store a file as a whole, HDFS breaks files into fixed-size blocks for several reasons.

Q2. Why is a block in HDFS so large?

=> A. Efficient Disk I/O:

Largers blocks result in fewer metadata operations (like opening / closing files) per unit of data, reducing overhead.

Reading or writing a large block at once improves disk I/O efficiency compared to smaller chunks.

DATE:

B. Reduced N/W overhead:
Data transfer involves n/w communication b/w DataNodes & clients. Larger blocks amortize the n/w overhead over more data, improving efficiency.
Smaller blocks might result in a higher percentage of overhead due to frequent data transfer initiation

C. Optimal for Sequential Access:
HDFS is designed for applications with large files & mostly sequential access patterns.
Larger blocks suit this design by allowing for continuous reading / writing of data, reducing seek time & improving throughput.

D. Minimized Metadata Overhead:
HDFS NameNode maintains metadata about files, & larger blocks mean fewer file records to manage.
Smaller blocks would lead to more metadata entries for the same amount of data, increasing the NameNode's load.

E. Better Load Distribution:
Large blocks help distribute data uniformly across DataNodes, preventing hotspots where certain nodes become overloaded due to small block sizes.