Getting Started with MongoDB

1. **Let's firstly understand what a NoSQL database is and why is it discussed so much nowadays?**

   https://www.mongodb.com/nosql-explained#:~:text=NoSQL%20databases%20(aka%20%22not%20only,wide%2Dcolumn%2C%20and%20graph.

2. **Now that we know what we are going to learn about, let's understand the query language MongoDB provides us to interact with the "DB"**

   Reference Video - MongoDB Crash Course

3. **I am also providing the cheat sheet as a go-to doc whenever we feel to revise MongoDB queries. In these queries, we are performing CRUD operations on a "posts" collection within the "acme" database.**

MongoDB Cheat Sheet

General Database handling

1. Show All Databases

- **show dbs**

2. Show Current Database

- **db**

3. Create Or Switch Database

- **use acme**

4. Drop

- **db.dropDatabase()**

5. Create Collection

- **db.createCollection('posts')**

6. Show Collections

- **show collections**

Inserting documents

7. Insert Row

- db.posts.insert({
  title: 'Post One',
  body: 'Body of post one',
  category: 'News',
  tags: ['news', 'events'],
  user: {
    name: 'John Doe',
    status: 'author'
  },
  date: Date()
})

8. Insert Multiple Rows

```
- db.posts.insertMany([
{
    title: 'Post Two',
    body: 'Body of post two',
    category: 'Technology',
    date: Date()
},
{
    title: 'Post Three',
    body: 'Body of post three',
    category: 'News',
    date: Date()
},
{
    title: 'Post Four',
    body: 'Body of post three',
    category: 'Entertainment',
  date: Date()
}
])
```

Retrieving Data -

9. Get All Rows

- **db.posts.find()**

10.Get All Rows Formatted

- **db.posts.find().pretty()**

11. Find Rows

- **db.posts.find({ category: 'News' })**

12. Sort Rows

 # asc

db.posts.find().sort({ title: 1 }).pretty()

 # desc

db.posts.find().sort({ title: -1 }).pretty()

13. Count Rows

```
db.posts.find().count()

db.posts.find({ category: 'news' }).count()
```

14. Limit Rows

```
db.posts.find().limit(2).pretty()
```

15. Chaining

```
db.posts.find().limit(2).sort({ title: 1 }).pretty()
```

16. Foreach

```
db.posts.find().forEach(function(doc) {
  print("Blog Post: " + doc.title)
})
```

17. Find One Row

```
db.posts.findOne({ category: 'News' })
```

18. Find Specific Fields

```
db.posts.find({ title: 'Post One' }, {
  title: 1,
  author: 1
})
```

Updating Documents -

19. Update Document

```
db.posts.update({ title: 'Post Two' },
{
  title: 'Post Two',
  body: 'New body for post 2',
  date: Date()
},
{
  upsert: true
})
```

20. Update Specific Field

```
db.posts.update({ title: 'Post Two' },
{
  $set: {
        body: 'Body for post 2',
        category: 'Technology'
  }
})
```

21. Increment Field ($inc)

```
db.posts.update({ title: 'Post Two' },
{
  $inc: {
        likes: 5
  }
})
```

22. Rename Field

```
db.posts.update({ title: 'Post Two' },
{
  $rename: {
        likes: 'views'
  }
})
```

Deleting Documents -

23. Delete Row

```
db.posts.remove({ title: 'Post Four' })
```

24. Sub-Documents

```
db.posts.update({ title: 'Post One' },
{
  $set: {
        comments: [
        {
        body: 'Comment One',
        user: 'Mary Williams',
        date: Date()
        },
        {
        body: 'Comment Two',
        user: 'Harry White',
        date: Date()
        }
        ]
  }
})
```

Some more important queries -

25. Find By Element in Array ($elemMatch)

```
db.posts.find({
  comments: {
        $elemMatch: {
        user: 'Mary Williams'
        }
        }
  }
)
```

26. Add Index

```
db.posts.createIndex({ title: 'text' })
```

27. Text Search

```
db.posts.find({
  $text: {
        $search: "\"Post O\""
        }
})
```

28. Greater & Less Than

```
db.posts.find({ views: { $gt: 2 } })
db.posts.find({ views: { $gte: 7 } })
db.posts.find({ views: { $lt: 7 } })
db.posts.find({ views: { $lte: 7 } })
```