**Question:**

Enterprise Z is developing a content management system that will handle a variety of material kinds such as “articles, photos, and videos”.

• Creating an effective data model to manage a variety of content kinds.

• Improving query performance for faster content retrieval.

• Ensuring data consistency across several content kinds.

• Adapting to changes in content structure.

**Solution:**

1. **Creating an effective data model to manage a variety of content.**
2. MongoDB is suited for handling various content types such as “articles, photos, and videos”.
3. Each record in collection represent a piece of content with its own structure.
4. We can use references to handle relationship between records.

Database name: Instagram\_contents

Collection name: content

Inserting documents: db.content.insertMany([{

\_id:1,

type: 'article',

title: 'MongoDB is NoSql',

content: 'MongoDB is suited for handling various contents',

tags: ['database', 'NoSQL',’MongoDB’]},

{

type: 'photo',

title: 'Ocean View',

imageURL: 'https://rise.com/ocean.jpg',

location: 'Ocean',

article\_id :1

},

{type: 'video',

title: 'Getting Started with MongoDB',

videoURL: 'https://udemy.com/mongodb.mp4',

duration: '11 minutes'}]);

db.content.insertOne({

type:'article',

title:'Java programming',

content:'Learn DSA using Java',

tags:['prgramming','java','DSA','Coding']})

image: 1.1 (create a new collection & inserting documents with linked contents)



Image 1.2 (insertOne document)

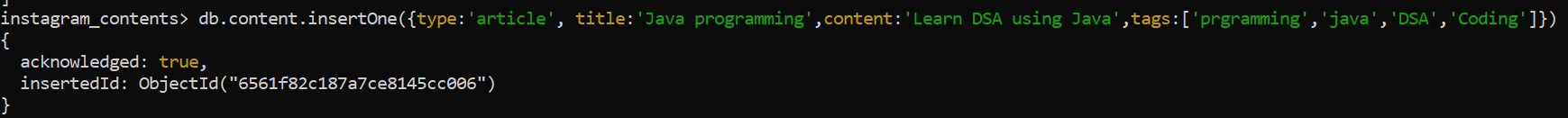


Image: 1.3 (Query documents in a collection)



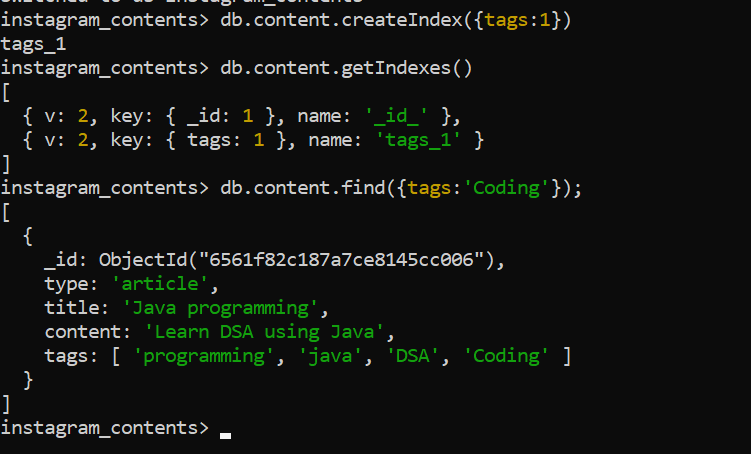
1. **Improving query performance for faster content retrieval.**

**Indexing** - MongoDB uses indexing in order to make the query processing more efficient. If there is no indexing, then the MongoDB must scan every document in the collection and retrieve only those documents that match the query.

Here, we can create index on **“tags”** key for faster retrieval.

**Creating Index:** db.content.createIndex({ tags: 1 });

Image: 2.1 (create indexing on a key)



1. **Ensuring data consistency across several content kinds.**

**ACID – A**tomicity, **C**onsistency, **I**solation and **D**urability

**Atomicity and Data Consistency:** When a **transaction consists of multiple operations, either all of these operations are executed successfully, or none of them are**. This prevents partial updates or incomplete transactions that could leave the database in an inconsistent state.

Atomicity is **essential for data consistency** because it ensures that a transaction is an all-or-nothing proposition. Either all the changes made by the transaction are applied successfully, and the database is left in a consistent state, or none of the changes are applied, and the database remains unchanged.

**Isolation:** Isolation ensures that **the execution of one transaction is isolated from the execution of other** concurrent transactions.

**Durability:** It ensures that once a transaction is committed, the changes made by **that transaction will persist even in the face of system failures** or crashes.

1. **Adapting to changes in content structure.**
2. MongoDB's flexible schema allows to adapt the changes easily.
3. We can add / remove fields to documents without affecting the entire collection.

Image: 4.1 (add new field to an existing document)

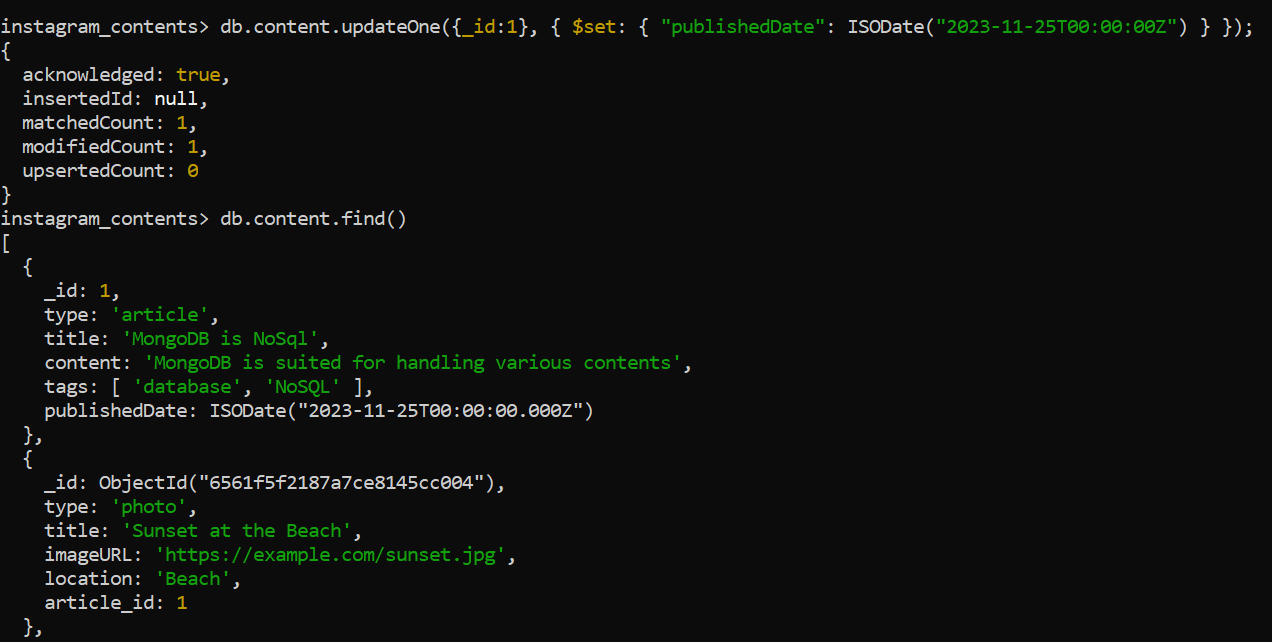


Image: 4.2 (remove a field from a document)

