Name: WAQAR RIASAT ALI Section: E Roll num: 2023F - BSE -221

LAB NO 11

PROJECTION AND DATA MODELING IN MONGODB

OBJECTIVE:

- To learn and implement projection in mongoDB.
- To learn and implement different types of data models in mongoDB.

LAB TASKS:

1. Study the following scenarios and suggest how the database should be modeled whether embedded or referenced? Support you answer with reasons.
   Scenario 1: Blogging Platform
   You are designing a MongoDB schema for a blogging platform. Each blog post can have multiple comments.
   Scenario 2: E-commerce Website
   You are building a database for an e-commerce website. Each product can have multiple reviews.
   Scenario 3: Geographic Data
   You are working on a system that stores geographic data, including countries, regions, and cities.
   Scenario 4: Messaging App
   You are developing a messaging application. Each conversation can have multiple messages.

Scenario 1: Blogging Platform

- **Model: Embedded**
- **Reason:** Each blog post contains its own comments. Comments are tightly coupled with posts and are rarely accessed independently.

Scenario 2: E-commerce Website

- **Model: Embedded**
- **Reason:** Reviews are generally read along with the product. Embedding reviews improves read performance.

Scenario 3: Geographic Data

- **Model: Referenced**
- **Reason:** Countries, regions, and cities are individual entities with many-to-one relationships. These are better normalized to avoid data duplication.

Scenario 4: Messaging App

- **Model: Embedded**
- **Reason:** Messages belong to a conversation and are typically loaded together. Embedding supports fast retrieval.

Name: WAQAR RIASAT ALI Section: E Roll num: 2023F - BSE -221

2. Create collections for scenarios in task#1 and insert documents (as per your answer in task#1).

Blogging Platform (Embedded)

```
test>     // Create the 'posts' collection
...     db.createCollection("posts")
...
...     // Insert a sample post with embedded comments
...     db.posts.insertOne({
...       title: "My Trip to Hunza Valley",
...       content: "Sharing my amazing experience in the beautiful Hunza Valley.",
...       author: "Fatima Khan",
...       comments: [
...         {
...           username: "AliRaza",
...           text: "Mashallah, beautiful!",
...           timestamp: new Date()
...         },
...         {
...           username: "SanaAhmed",
...           text: "Incredible scenery.  Thanks for sharing!",
...           timestamp: new Date()
...         }
...       ]
...     })
...
{
  acknowledged: true,
  insertedId: ObjectId('684808a01bfd40751850eb67')
}
```

E-commerce Website (Embedded)

```
test>     // Create the 'products' collection
...     db.createCollection("products")
...
...     // Insert a sample product with embedded reviews
...     db.products.insertOne({
...       name: "Khaadi Lawn Suit",
...       description: "Elegant Khaadi lawn suit, perfect for summer.",
...       price: 4500.00, // Pakistani Rupees
...       reviews: [
...         {
...           username: "AyeshaKhan",
...           rating: 5,
...           comment: "Excellent quality and design!"
...         },
...         {
...           username: "HinaButt",
...           rating: 4,
...           comment: "Good value for money."
...         }
...       ]
...     })
...
{
  acknowledged: true,
  insertedId: ObjectId('684808b01bfd40751850eb68')
}
```

Name: WAQAR RIASAT ALI Section: E Roll num: 2023F - BSE -221

Geographic Data (Referenced)

```
test>      // Create the 'countries' collection
...      db.createCollection("countries")
...
...      // Insert a sample country
...      db.countries.insertOne({
...        name: "Pakistan",
...        code: "PK"
...      })
...
...      // Create the 'regions' collection
...      db.createCollection("regions")
...
...      // Insert a sample region, referencing the country
...      const pakistan = db.countries.findOne({name: "Pakistan"})
...      db.regions.insertOne({
...        name: "Punjab",
...        country_id: pakistan._id // Reference to the Pakistan document
...      })
...
...      // Create the 'cities' collection
...      db.createCollection("cities")
...
...      // Insert a sample city, referencing the region
...      const punjab = db.regions.findOne({name: "Punjab"})
...      db.cities.insertOne({
...        name: "Lahore",
...        region_id: punjab._id // Reference to the Punjab document
...      })
...
{
  acknowledged: true,
  insertedId: ObjectId('684808c81bfd40751850eb6b')
}
```

Messaging App (Embedded)

```
test>      // Create the 'conversations' collection
...      db.createCollection("conversations")
...
...      // Insert a sample conversation with embedded messages
...      db.conversations.insertOne({
...        participants: ["Ahmed123", "ZaraKhan"],
...        messages: [
...          {
...            sender: "Ahmed123",
...            text: "Assalam-o-Alaikum! Kya haal hai?",
...            timestamp: new Date()
...          },
...          {
...            sender: "ZaraKhan",
...            text: "Walaikum Assalam!  Theek thaak. Aap sunao?",
...            timestamp: new Date()
...          }
...        ]
...      })
...
{
  acknowledged: true,
  insertedId: ObjectId('684808d71bfd40751850eb6c')
}
```

3. Apply retrieval operation in the collections created above, and show how the data is retrieved in referenced model as compared to embedded model (utilize projection).

Embedded Model (Blogging Platform):

```
test>     // Retrieve a blog post and project only the title and comments
...     db.posts.find(
...       { title: "My First Blog Post" },
...       { title: 1, comments: 1, _id: 0 } // Projection: include title and comments, exclude _id
...     ).pretty()
```

Referenced Model (Geographic Data):

```
test>     // Retrieve city information along with its region and country
...     db.cities.aggregate([
...       {
...         $lookup: {
...           from: "regions",
...           localField: "region_id",
...           foreignField: "_id",
...           as: "region"
...         }
...       },
...       {
...         $unwind: "$region"
...       },
...       {
...         $lookup: {
...           from: "countries",
...           localField: "region.country_id",
...           foreignField: "_id",
...           as: "country"
...         }
...       },
...       {
...         $unwind: "$country"
...       },
...       {
...         $project: {  //Projection to shape the output
...           "city_name": "$name",
...           "region_name": "$region.name",
...           "country_name": "$country.name",
...           "_id": 0
...         }
...       }
...     ]).pretty()
[
  {
    city_name: 'Lahore',
    region_name: 'Punjab',
    country_name: 'Pakistan'
  },
  {
    city_name: 'Lahore',
    region_name: 'Punjab',
    country_name: 'Pakistan'
  }
]
```

4. Explore and implement $elematch and $regex methods and embed them in your program (in any of the above scenarios).

```
test>      db.posts.find(
...        {
...          comments: {
...            $elemMatch: {
...              username: "Jane",
...              rating: { $gt: 3 }
...            }
...          }
...        }
...      ).pretty()
...
test>     db.posts.find(
...       {
...         "comments.text": { $regex: "great|love", $options: "i" }
...       }
...     ).pretty()
```

5. Answer the following questions: • What have you learned from the lab tasks? • What was the most challenging task and how did you overcome that challenge?

What have you learned from the lab task?

- Select specific fields for efficient queries.
- Embedding vs. referencing trade-offs (read speed vs. data normalization).
- Querying array elements with multiple criteria.
- Flexible pattern matching in strings.
- Joining collections and reshaping data.
- Adapting examples to a specific culture (Pakistan).

What was the most challenging task and how did you overcome it?

- Deciding between embedding and referencing, mastering aggregation.
- Analyzed relationships, weighed trade-offs, practiced aggregation framework, consulted documentation.