**Q1. What is MongoDB?**

**MongoDB** is a NoSQL database that stores data in a flexible, JSON-like format called BSON (Binary JSON). Unlike traditional relational databases, MongoDB does not require a predefined schema, which allows for easier handling of unstructured data and rapid iteration in application development. It is designed for scalability, high availability, and performance, making it suitable for handling large amounts of data across distributed systems.

**Q2. What is the difference between MongoDB and SQL?**

1. **Data Model**:
   * **MongoDB**: Document-based, stores data in BSON (similar to JSON) format. Each document can have a different structure.
   * **SQL (Relational Databases)**: Table-based, stores data in rows and columns. Schema must be defined before storing data.
2. **Schema**:
   * **MongoDB**: Schema-less, allowing for flexible and dynamic data structures.
   * **SQL**: Schema-bound, requiring predefined structure for tables, columns, and data types.
3. **Scalability**:
   * **MongoDB**: Horizontally scalable through sharding (distributing data across multiple servers).
   * **SQL**: Typically vertically scalable (adding more resources to a single server).
4. **Relationships**:
   * **MongoDB**: Supports embedded documents and references for relationships, but does not enforce foreign keys.
   * **SQL**: Supports complex relationships between tables using foreign keys and JOIN operations.
5. **Query Language**:
   * **MongoDB**: Uses a query language based on JSON-like syntax, allowing for powerful querying capabilities.
   * **SQL**: Uses SQL (Structured Query Language), a standard language for querying and managing relational databases.

**Q3. Create a Database for an Online Shopping App**

To create a database named OnlineShoppingApp in MongoDB, you would use the following command:

use OnlineShoppingApp

**Q4. Create Required Collections for the Online Shopping App and Documents**

**i. User Collection**

db.createCollection("User")

db.User.insertMany([

{

"\_id": ObjectId("user\_id\_1"),

"username": "john\_doe",

"email": "john@example.com",

"password": "hashed\_password",

"address": "123 Main St, City, Country",

"phone": "123-456-7890",

"created\_at": new Date()

},

{

"\_id": ObjectId("user\_id\_2"),

"username": "jane\_doe",

"email": "jane@example.com",

"password": "hashed\_password",

"address": "456 Elm St, City, Country",

"phone": "987-654-3210",

"created\_at": new Date()

}

])

**ii. Product Category Collection**

db.createCollection("ProductCategory")

db.ProductCategory.insertMany([

{

"\_id": ObjectId("category\_id\_1"),

"name": "Electronics",

"description": "Electronic gadgets and devices"

},

{

"\_id": ObjectId("category\_id\_2"),

"name": "Clothing",

"description": "Apparel and accessories"

}

])

**iii. Product Collection**

db.createCollection("Product")

db.Product.insertMany([

{

"\_id": ObjectId("product\_id\_1"),

"name": "Smartphone",

"category\_id": ObjectId("category\_id\_1"),

"price": 699.99,

"stock": 50,

"description": "Latest model smartphone with high-end features",

"created\_at": new Date()

},

{

"\_id": ObjectId("product\_id\_2"),

"name": "Jeans",

"category\_id": ObjectId("category\_id\_2"),

"price": 49.99,

"stock": 100,

"description": "Comfortable and stylish denim jeans",

"created\_at": new Date()

}

])

**iv. Order Collection**

db.createCollection("Order")

db.Order.insertMany([

{

"\_id": ObjectId("order\_id\_1"),

"user\_id": ObjectId("user\_id\_1"),

"product\_ids": [ObjectId("product\_id\_1")],

"total\_amount": 699.99,

"status": "Processing",

"created\_at": new Date()

},

{

"\_id": ObjectId("order\_id\_2"),

"user\_id": ObjectId("user\_id\_2"),

"product\_ids": [ObjectId("product\_id\_2")],

"total\_amount": 49.99,

"status": "Shipped",

"created\_at": new Date()

}

])

**v. Review Collection**

db.createCollection("Review")

db.Review.insertMany([

{

"\_id": ObjectId("review\_id\_1"),

"product\_id": ObjectId("product\_id\_1"),

"user\_id": ObjectId("user\_id\_1"),

"rating": 5,

"comment": "Excellent smartphone with great features",

"created\_at": new Date()

},

{

"\_id": ObjectId("review\_id\_2"),

"product\_id": ObjectId("product\_id\_2"),

"user\_id": ObjectId("user\_id\_2"),

"rating": 4,

"comment": "Good quality jeans, comfortable fit",

"created\_at": new Date()

}

])

**Q5. Write Command to Show All Data from Product Collection and Sort in Ascending Order**

db.Product.find().sort({ "name": 1 })

This command finds all documents in the Product collection and sorts them by the name field in ascending order (1 means ascending, -1 means descending).

**Q6. Update Product Price for a Particular Product**

For example, to update the price of the product with product\_id\_1 to 749.99:

db.Product.updateOne(

{ "\_id": ObjectId("product\_id\_1") },

{ $set: { "price": 749.99 } }

)

**Q7. Write Command to Delete a Particular Document and Collection**

**Delete a Particular Document**

To delete a specific document, for example, the product with product\_id\_2:

db.Product.deleteOne({ "\_id": ObjectId("product\_id\_2") })

**Delete an Entire Collection**

To delete the Product collection:

db.Product.drop()

These commands will help you manage and manipulate the MongoDB database for your online shopping app.

Top of Form

Bottom of Form