**🔹 ACID Properties of a Transaction:**

| **ACID Property** | **Description** | **Example** |
| --- | --- | --- |
| **Atomicity** | **All steps succeed or all fail.** | **₹500 transfer: deducted from A and added to B, or no change if fails.** |
| **Consistency** | **Maintains valid data and follows rules.** | **Stock can’t go below zero; transaction rejected if it tries.** |
| **Isolation** | **Transactions don’t interfere with each other.** | **Two users booking the last seat: only one booking succeeds.** |
| **Durability** | **Committed changes are permanent.** | **Movie ticket booking remains saved even if server crashes.** |

# **📚 MongoDB Learning Notes (Full Summary)**

## **📦 Why MongoDB and Not Files?**

* **Files like CSV, JSON, or plain text are:**
  + **Hard to search, update, and scale.**
  + **Lacking in querying capabilities.**
* **MongoDB advantages:**
  + **Flexible schema with JSON-like documents.**
  + **Indexed, searchable, and scalable.**
  + **Real-time data access and updates.**

## **🛠️ CRUD Operations in MongoDB**

### **📅 Insert Operations**

**✅ Insert one document:**

**db.users.insertOne({**

**name: "Alice",**

**age: 25,**

**email: "alice@example.com",**

**status: "pending"**

**})**

**✅ Insert multiple documents:**

**db.users.insertMany([**

**{ name: "Bob", age: 32, email: "bob@example.com", status: "pending" },**

**{ name: "Charlie", age: 37, email: "charlie@gmail.com", status: "pending" }**

**])**

### **🔍 Read (Query) Operations**

**✅ Find all:**

**db.users.find()**

**✅ Pretty print:**

**db.users.find().pretty()**

**✅ Projection (include name and email, exclude \_id):**

**db.users.find(**

**{ status: "pending" },**

**{ name: 1, email: 1, \_id: 0 }**

**)**

**✅ Examples using conditions:**

**// Age >= 25 and active**

**db.users.find({ age: { $gte: 25 }, status: "active" })**

**// Find users named Fiona or Jane**

**db.users.find({ $or: [{ name: "Fiona" }, { name: "Jane" }] })**

**// Email ends with "@example.com"**

**db.users.find({ email: { $regex: /@example\.com$/ } })**

**// Age < 30 or status = "pending"**

**db.users.find({**

**$or: [**

**{ age: { $lt: 30 } },**

**{ status: { $eq: "pending" } }**

**]**

**})**

### **✏️ Update Operations**

**✅ Update one document:**

**db.users.updateOne(**

**{ name: "Alice" },**

**{ $set: { status: "active" } }**

**)**

**✅ Increment age of all users:**

**db.users.updateMany({}, { $inc: { age: 1 } })**

**✅ Remove a field:**

**db.users.updateOne(**

**{ name: "Bob" },**

**{ $unset: { email: "" } }**

**)**

**✅ Replace a document:**

**db.users.replaceOne(**

**{ name: "Charlie" },**

**{**

**name: "Charlie",**

**age: 40,**

**email: "charlie@newmail.com",**

**status: "active"**

**}**

**)**

### **🗑️ Delete Operations**

**✅ Delete one:**

**db.users.deleteOne({ name: "Alice" })**

**✅ Delete many:**

**db.users.deleteMany({ status: "pending" })**

**✅ Delete if conditionally matched (example only):**

**db.users.remove({ age: { $gte: 150 } })**

## **🔍 Query Operators Summary**

### **📌 Comparison Operators**

| **Operator** | **Meaning** | **Example** |
| --- | --- | --- |
| **$eq** | **Equals** | **{ age: { $eq: 30 } }** |
| **$ne** | **Not equals** | **{ status: { $ne: "active" } }** |
| **$gt** | **Greater than** | **{ age: { $gt: 25 } }** |
| **$gte** | **Greater than or equal** | **{ age: { $gte: 18 } }** |
| **$lt** | **Less than** | **{ age: { $lt: 40 } }** |
| **$lte** | **Less than or equal** | **{ age: { $lte: 50 } }** |
| **$in** | **In array** | **{ status: { $in: ["active"] }}** |
| **$nin** | **Not in array** | **{ name: { $nin: ["Alice"] }}** |

### **🔗 Logical Operators**

**$and: [ { age: { $gte: 18 } }, { status: "active" } ]**

**$or : [ { age: { $lt: 18 } }, { status: "pending" } ]**

**$not: { age: { $not: { $gt: 30 } } }**

### **🔠 Regex Operators**

**✅ Find users with email ending in @example.com:**

**{ email: { $regex: /@example\.com$/ } }**

**Regex Tips:**

* **^ → starts with**
* **$ → ends with**
* **. → any character (escaped as \.)**

## **✅ Tips for Mongo Shell**

* **Use .pretty() to display output cleanly.**
* **Use MongoDB Compass or Atlas UI for GUI-based interactions.**
* **Double-check conditions before deleteMany() or remove().**
* **Use $set, $inc, $unset smartly for efficient updates.**