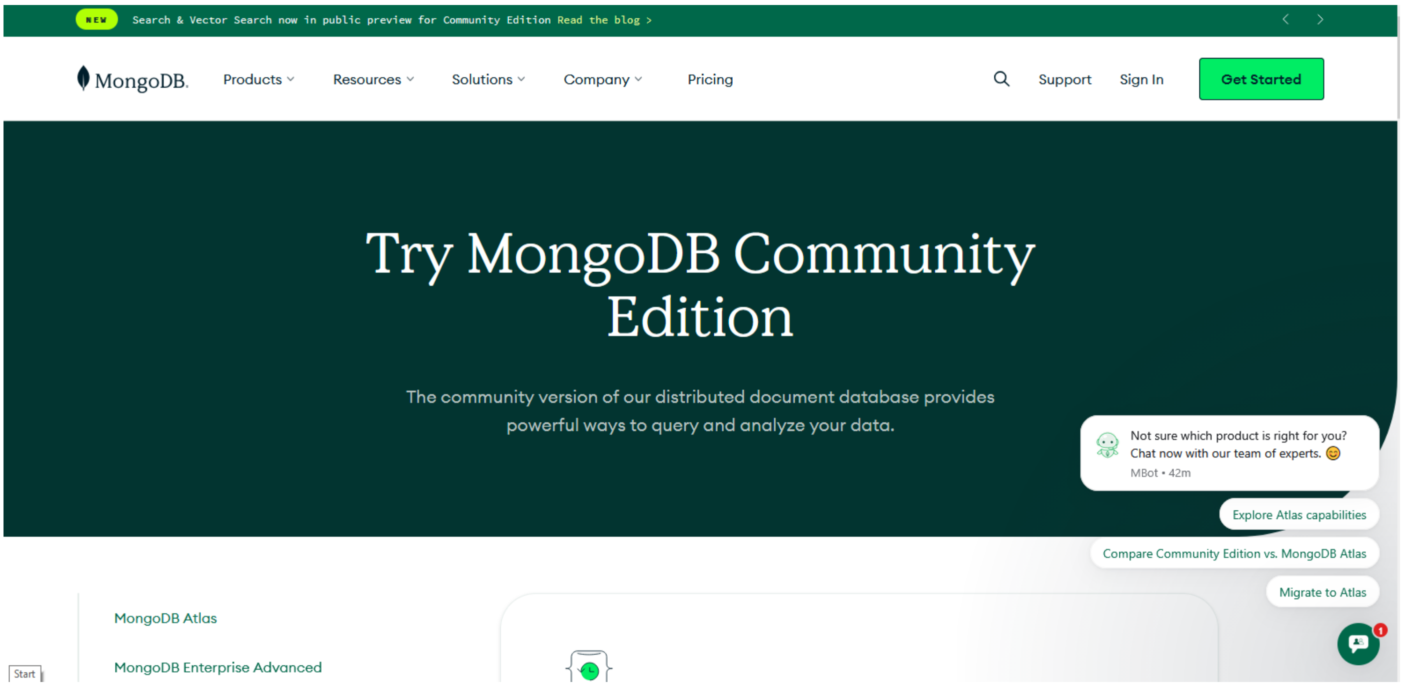
MongoDB Guide

- Emam Saimon

QUICK SETUP:

STEP	ACTION
1	Go to: https://www.mongodb.com/try/download/community
2	Choose **Windows/macOS/Linux** → Download
3	Install (default path is fine)
4	**Compass** is included in the installer (tick it)
5	Open **MongoDB Compass** → It auto-detects local server



NEW Search & Vector Search now in public preview for Community Edition Read the blog >

MongoDB. Products Resources Solutions Company Pricing Support Sign In Get Started

Try MongoDB Community Edition

The community version of our distributed document database provides powerful ways to query and analyze your data.

Not sure which product is right for you? Chat now with our team of experts. 😊

MBot • 42m

Explore Atlas capabilities

Compare Community Edition vs. MongoDB Atlas

Migrate to Atlas

MongoDB Atlas

MongoDB Enterprise Advanced

Your local MongoDB is running on mongodb://localhost:27027 (default port)

PART 1: MongoDB CONCEPTS – A to Z (Simple Words)

Letter	Concept	Easy Explanation
A	**Atlas**	MongoDB in the cloud (free tier available)
B	**BSON**	Binary JSON – how MongoDB stores data
C	**Collection**	Like a table in SQL, but **no fixed schema**
D	**Document**	One record = JSON-like object `{name: "Rahim", age: 25}`
E	**Embedded Documents**	Nesting: `{name: "Karim", address: {city: "Dhaka"}}`
F	**Field**	Key in a document (e.g., `age`)
G	**GridFS**	Store big files (>16MB) like videos
H	**_id**	Auto-generated unique ID (`ObjectId`)
I	**Index**	Speeds up queries (like book index)
J	**JSON**	Human-readable format MongoDB uses
K	**Key**	Same as Field
L	**Limit**	Show only N results
M	**MongoDB Shell (`mongosh`)**	Command line to talk to DB
N	**Namespace**	`database.collection`
O	**ObjectId**	12-byte unique ID with timestamp
P	**Projection**	Pick only some fields in result
Q	**Query**	Find data (`db.users.find({age: 25})`)
R	**Replica Set**	Backup copies for high availability
S	**Schema**	Flexible – no strict columns
T	**TTL Index**	Auto-delete docs after time
U	**Update**	Change existing docs
V	**Validation**	Enforce rules on insert/update
W	**Write Concern**	How many nodes must confirm write
X	**Explain**	See query performance
Y	**Aggregation**	Group, sum, filter like SQL `GROUP BY`
Z	**Zones (Sharding)**	Split data across servers

macOS (Homebrew)

```
brew tap mongodb/brew
brew install mongodb-community
```

Ubuntu/Debian

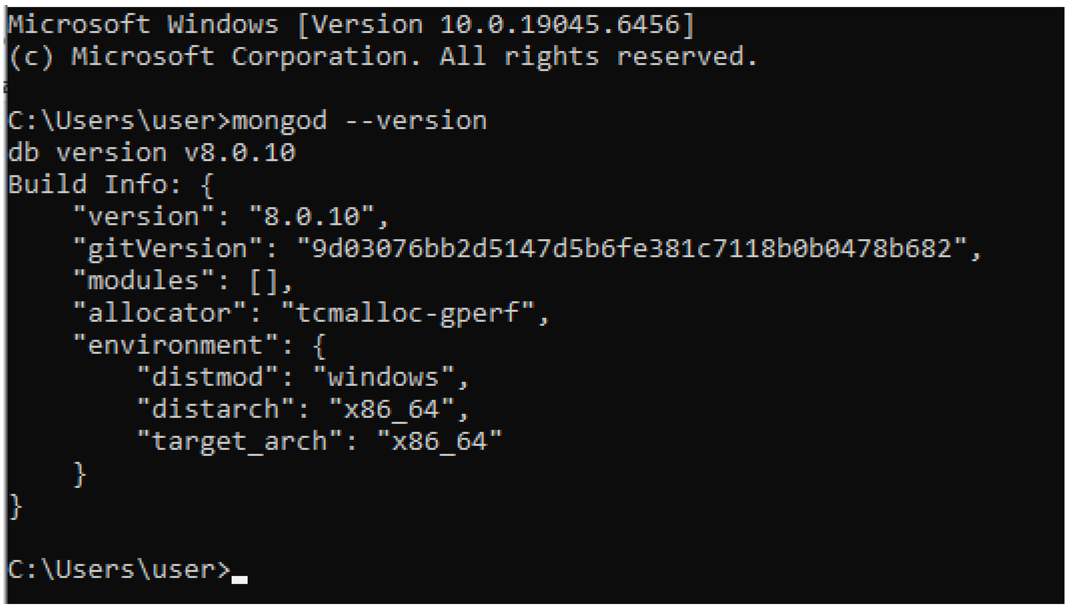
```
sudo apt-get install -y mongodb-org
```

Check Version & Start Server

1. Open Command Prompt

Press:
Win + R → type `cmd` → Enter

```
mongod --version

//or

mongo --version
```

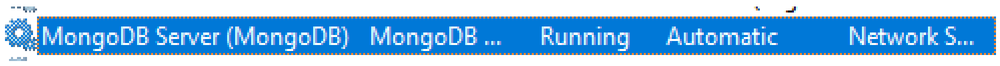


```
Microsoft Windows [Version 10.0.19045.6456]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user>mongod --version
db version v8.0.10
Build Info: {
    "version": "8.0.10",
    "gitVersion": "9d03076bb2d5147d5b6fe381c7118b0b0478b682",
    "modules": [],
    "allocator": "tcmalloc-gperf",
    "environment": {
        "distmod": "windows",
        "distarch": "x86_64",
        "target_arch": "x86_64"
    }
}

C:\Users\user>_
```

Check MongoDB Service (Server running or not)

services.msc

Find **MongoDB Server**
If it exists → MongoDB is installed.



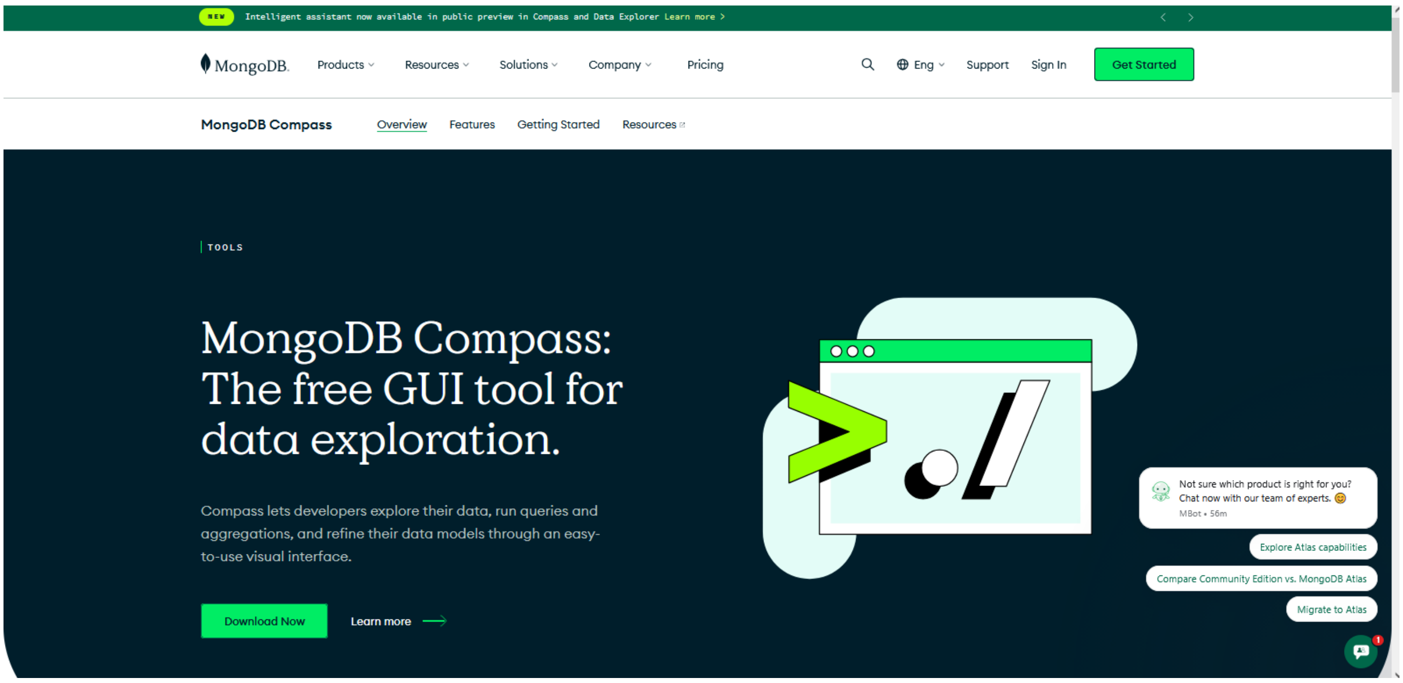
MongoDB Server (MongoDB) MongoDB ... Running Automatic Network S...

MongoDB Compass – The GUI Tool

1. **Open MongoDB Compass** (installed with MongoDB or download separately).
2. Click **"Connect"** to mongodb://localhost:27017.
3. You'll see:
    - List of databases
    - Collections (like tables)
    - Documents (like rows)

Compass CRUD (Easy Way)

Action	How in Compass
Create DB	Click **"Create Database"**
Insert Doc	Click collection → **ADD DATA** → **Insert Document**
Read	Click collection → see list
Update	Click pencil icon
Delete	Click trash icon



Intelligent assistant now available in public preview in Compass and Data Explorer

MongoDB Compass: The free GUI tool for data exploration.

Compass lets developers explore their data, run queries and aggregations, and refine their data models through an easy-to-use visual interface.

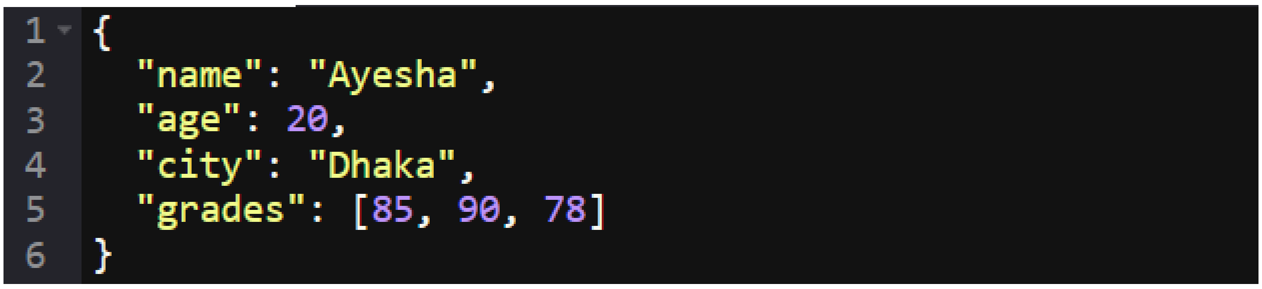
Not sure which product is right for you? Chat now with our team of experts. 😊
MBot • 56m

Create Database & Collection

1. Click **+ Create Database**
    - DB Name: school
    - Collection: students
2. Click **Create**

Insert Document (GUI)

- Click students → **INSERT DOCUMENT**



```
{
  "name": "Ayesha",
  "age": 20,
  "city": "Dhaka",
  "grades": [85, 90, 78]
}
```

Query Visually

- Filter box: { "age": { "$gt": 18 } }
- Projection: name, age

5. Aggregation Pipeline (GUI)

1. Click **Aggregations** tab
2. Add stage: $match → { age: { $gte: 20 } }
3. Add stage: $group → _id: "$city", count: { $sum: 1 }

Mongo Shell (mongosh) – ALL COMMANDS (Easy!)

Open terminal → type mongosh

A. Database Commands

Command	Meaning	Example
`use dbname`	Switch/create DB	`use school`
`show dbs`	List databases	`show dbs`
`db`	Current DB	`db → school`
`db.dropDatabase()`	Delete current DB	`db.dropDatabase()`

B. Collection Commands

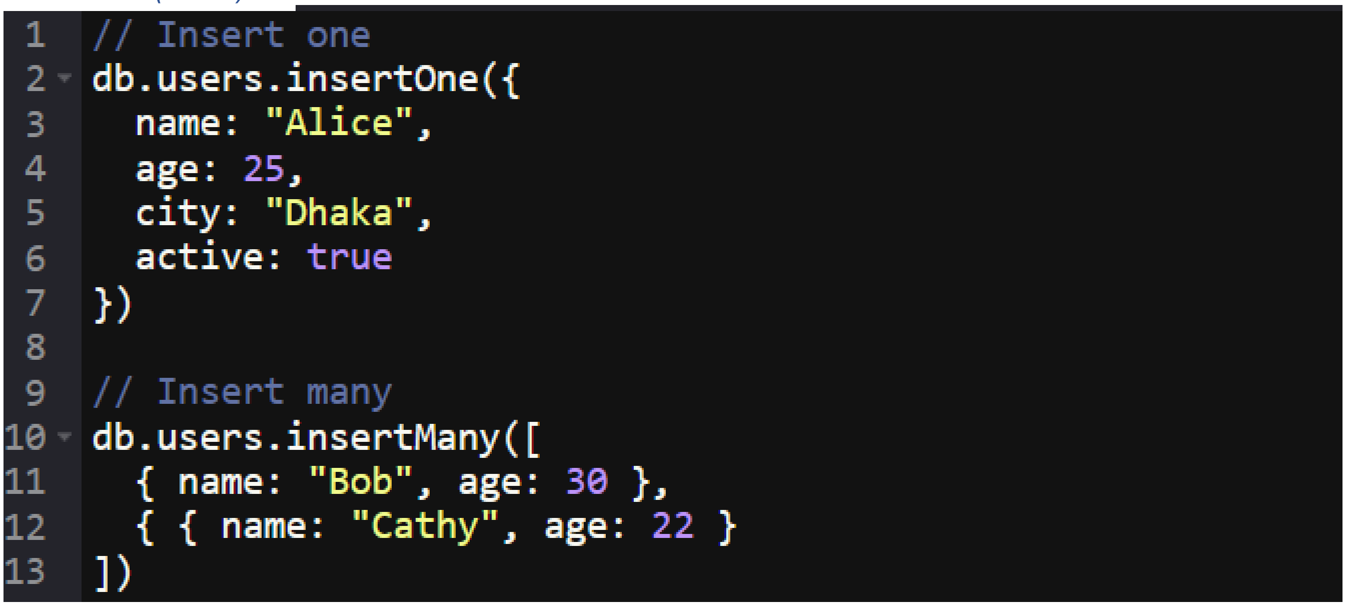
Command	Meaning	Example
`db.createCollection("name")`	Make collection	`db.createCollection("teachers")`
`show collections`	List collections	`show collections`
`db.collectioname.drop()`	Delete collection	`db.students.drop()`

Basic Command:

Command	Meaning
`show dbs`	List databases
`use mydb`	Switch/create DB
`show collections`	List collections
`db`	Current database

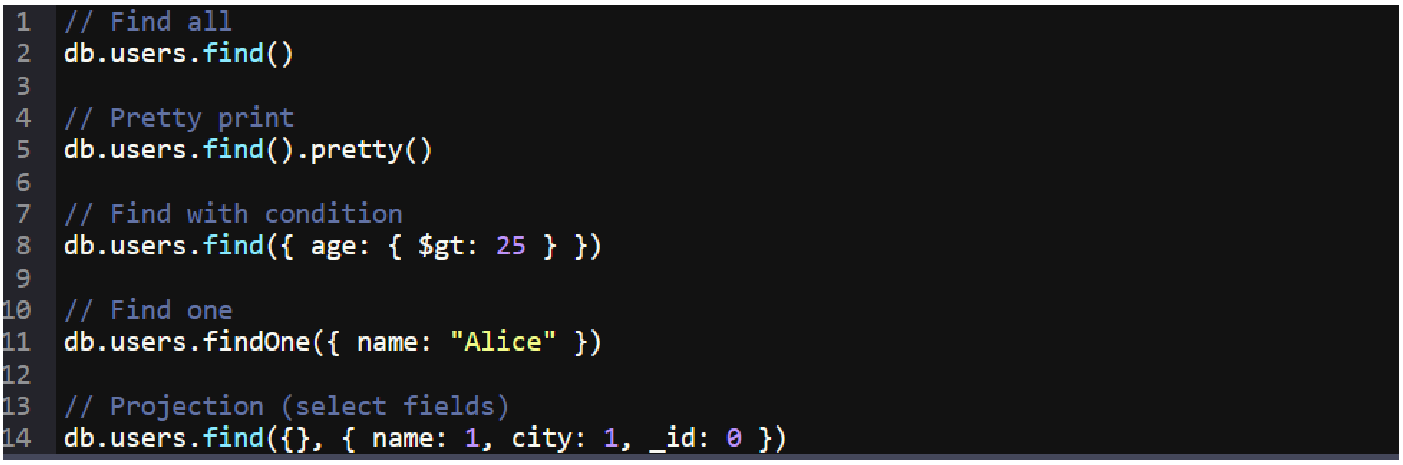
CRUD Operations (Pro Level)

C – Create (Insert)



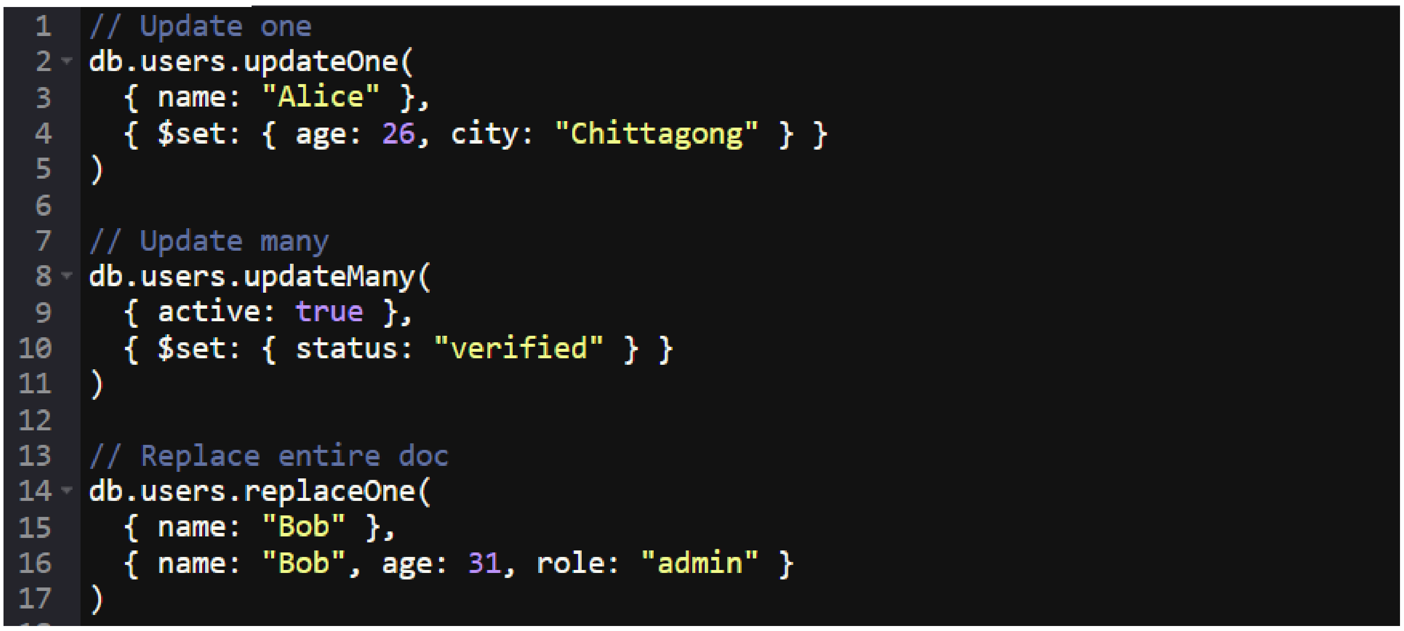
```
// Insert one
db.users.insertOne({
  name: "Alice",
  age: 25,
  city: "Dhaka",
  active: true
})

// Insert many
db.users.insertMany([
  { name: "Bob", age: 30 },
  { { name: "Cathy", age: 22 }
])
```

R – Read (Find)



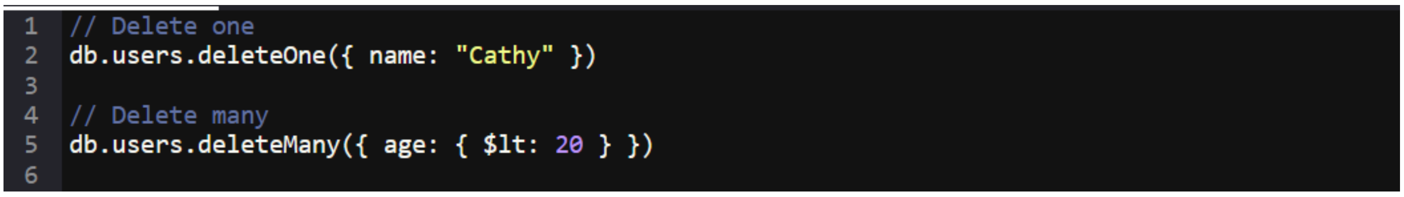
```
// Find all
db.users.find()

// Pretty print
db.users.find().pretty()

// Find with condition
db.users.find({ age: { $gt: 25 } })

// Find one
db.users.findOne({ name: "Alice" })

// Projection (select fields)
db.users.find({}, { name: 1, city: 1, _id: 0 })
```

U – Update



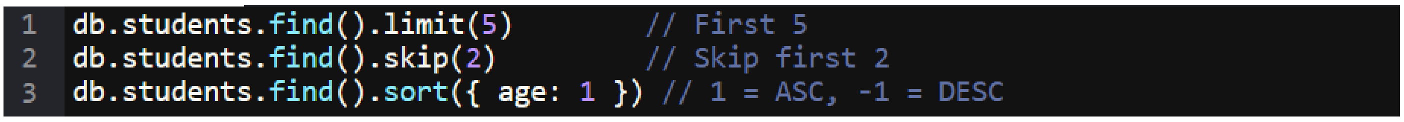
```
// Update one
db.users.updateOne(
  { name: "Alice" },
  { $set: { age: 26, city: "Chittagong" } }
)

// Update many
db.users.updateMany(
  { active: true },
  { $set: { status: "verified" } }
)

// Replace entire doc
db.users.replaceOne(
  { name: "Bob" },
  { name: "Bob", age: 31, role: "admin" }
)
```

D – Delete



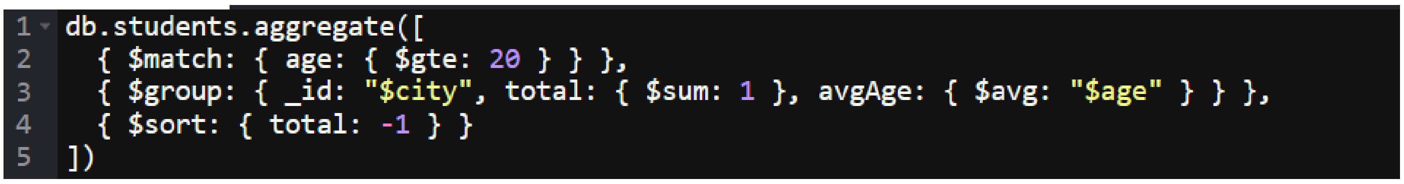
```
// Delete one
db.users.deleteOne({ name: "Cathy" })

// Delete many
db.users.deleteMany({ age: { $lt: 20 } })
```

LIMIT, SKIP, SORT



```
db.students.find().limit(5)        // First 5
db.students.find().skip(2)         // Skip first 2
db.students.find().sort({ age: 1 }) // 1 = ASC, -1 = DESC
```

AGGREGATION (Like SQL GROUP BY)

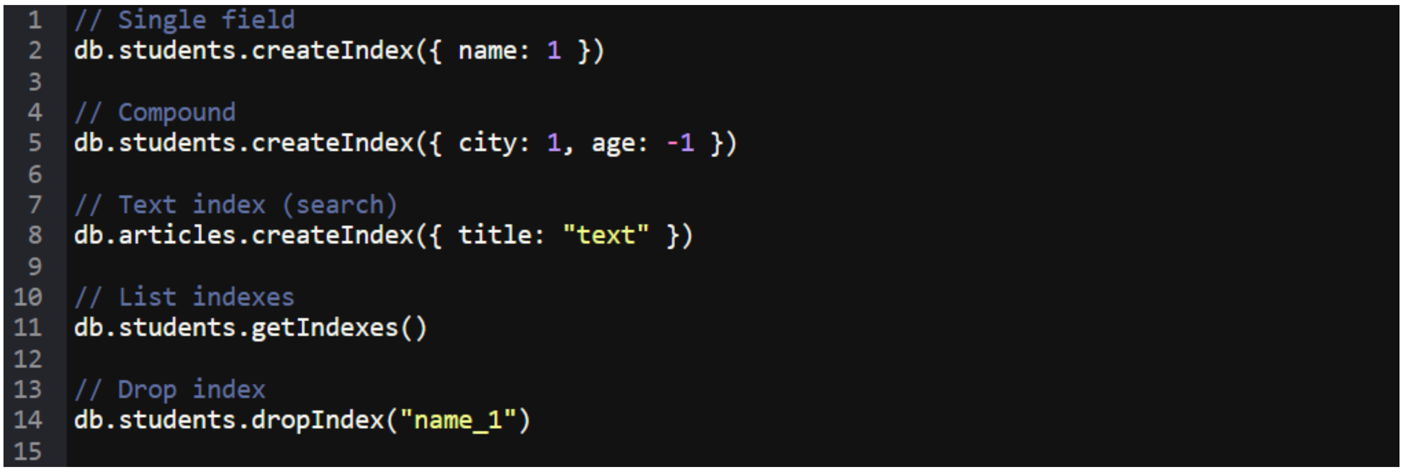


```
db.students.aggregate([
  { $match: { age: { $gte: 20 } } },
  { $group: { _id: "$city", total: { $sum: 1 }, avgAge: { $avg: "$age" } } },
  { $sort: { total: -1 } }
])
```

Aggregation Stages

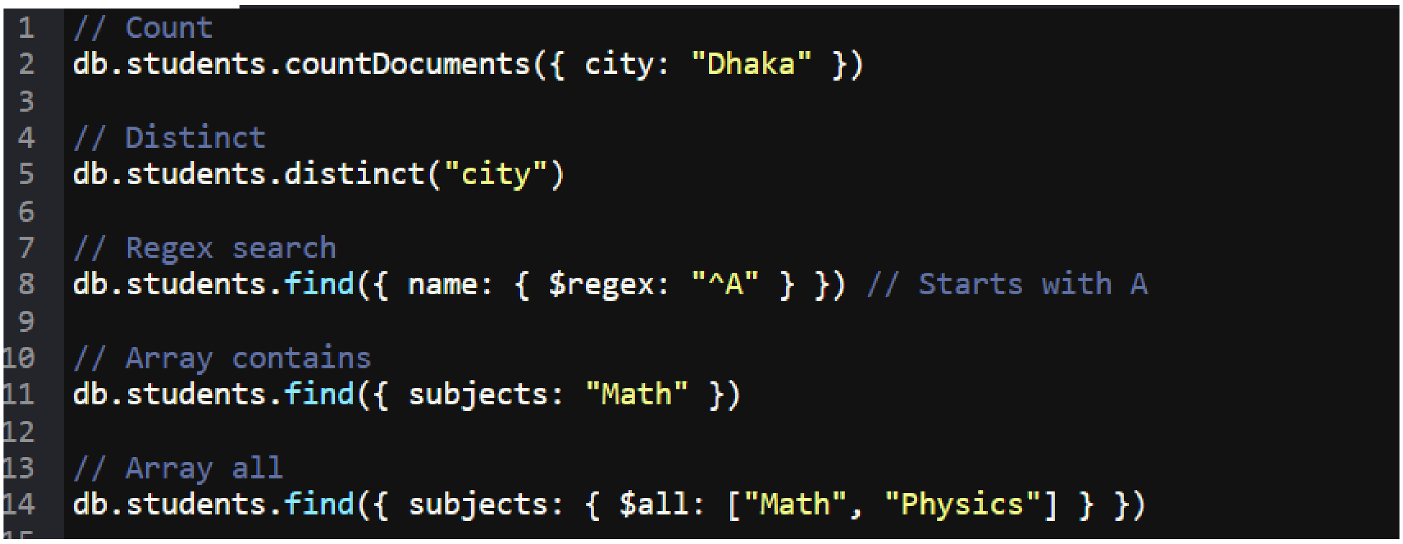
Stage	Meaning
`$match`	Filter
`$group`	Group + calculate
`$project`	Select/reshape
`$sort`	Order
`$limit`	Top N
`$unwind`	Expand array

INDEXES (Speed Up Queries)



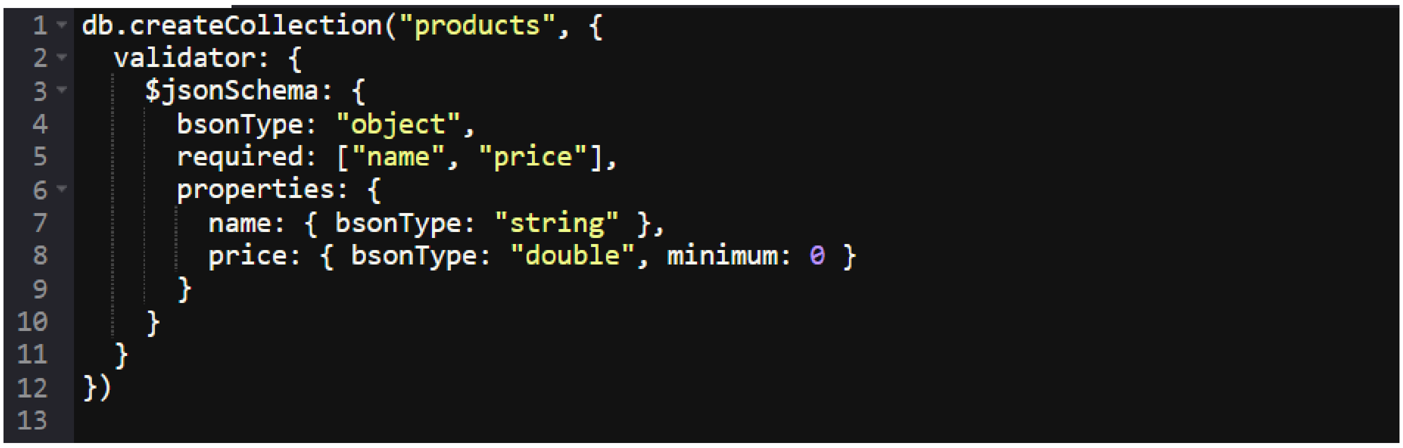
```
// Single field
db.students.createIndex({ name: 1 })

// Compound
db.students.createIndex({ city: 1, age: -1 })

// Text index (search)
db.articles.createIndex({ title: "text" })

// List indexes
db.students.getIndexes()

// Drop index
db.students.dropIndex("name_1")

```

SPECIAL QUERIES



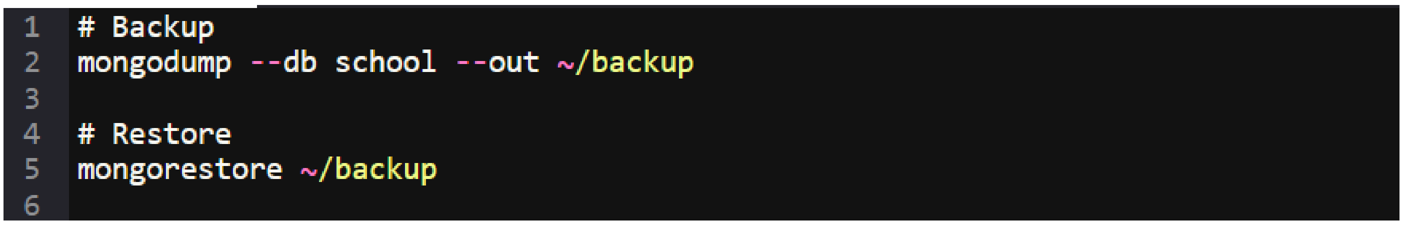
```
// Count
db.students.countDocuments({ city: "Dhaka" })

// Distinct
db.students.distinct("city")

// Regex search
db.students.find({ name: { $regex: "^A" } }) // Starts with A

// Array contains
db.students.find({ subjects: "Math" })

// Array all
db.students.find({ subjects: { $all: ["Math", "Physics"] } })
```

VALIDATION (Enforce Rules)



```
db.createCollection("products", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["name", "price"],
      properties: {
        name: { bsonType: "string" },
        price: { bsonType: "double", minimum: 0 }
      }
    }
  }
})
```

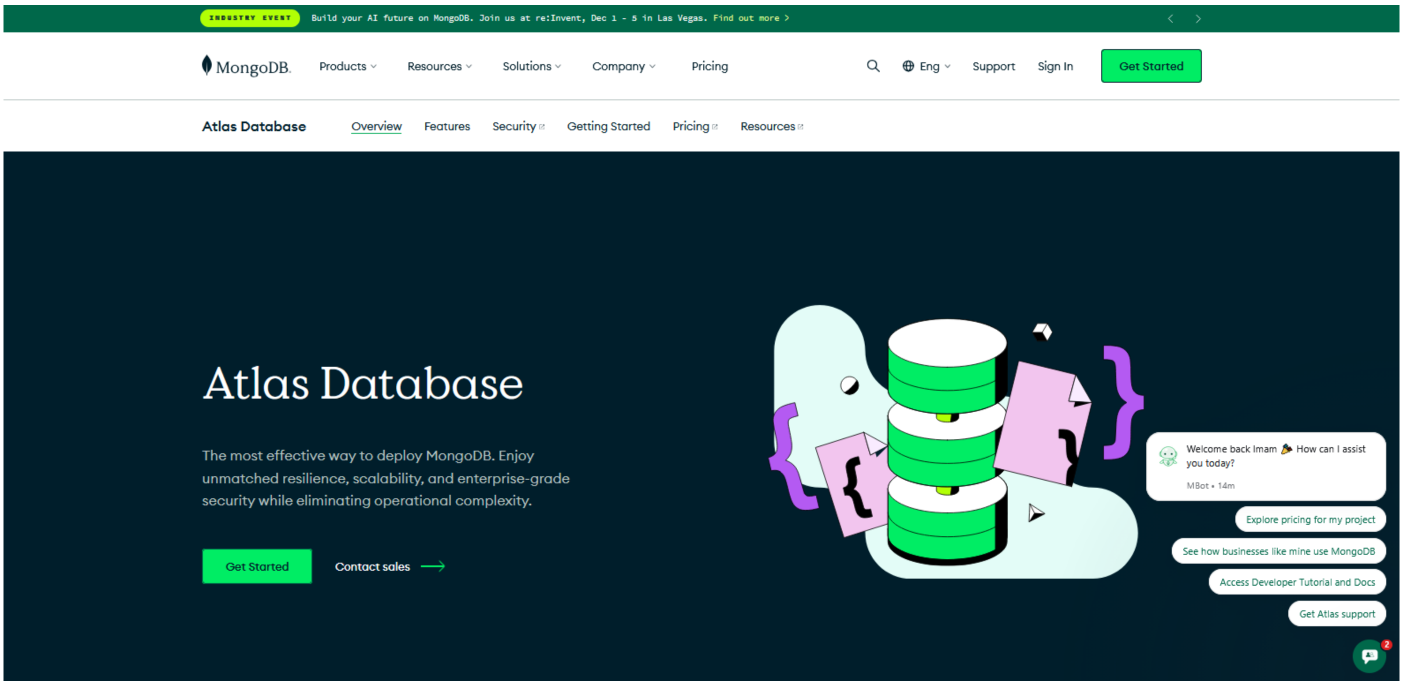
BACKUP & RESTORE (Local)



```
# Backup
mongodump --db school --out ~/backup

# Restore
mongorestore ~/backup
```

Pro Shell Commands (A to Z Cheat Sheet)

Command	Use
`db.collection.aggregate([...])`	Powerful data pipelines
`db.collection.createIndex({ field: 1 })`	Speed up queries
`db.collection.drop()`	Delete collection
`db.dropDatabase()`	Delete entire DB
`db.stats()`	DB size & stats
`db.collection.countDocuments({})`	Count docs
`db.collection.find().limit(5)`	Limit results
`db.collection.find().sort({ age: -1 })`	Sort descending
`db.collection.distinct("city")`	Unique values
`db.runCommand({ collMod: "users", validator: { ... } })`	Schema validation



Build your AI future on MongoDB. Join us at re:Invent, Dec 1 - 5 in Las Vegas. Find out more >

Atlas Database

The most effective way to deploy MongoDB. Enjoy unmatched resilience, scalability, and enterprise-grade security while eliminating operational complexity.

Welcome back Imam 🎉 How can I assist you today?

MBot • 14m

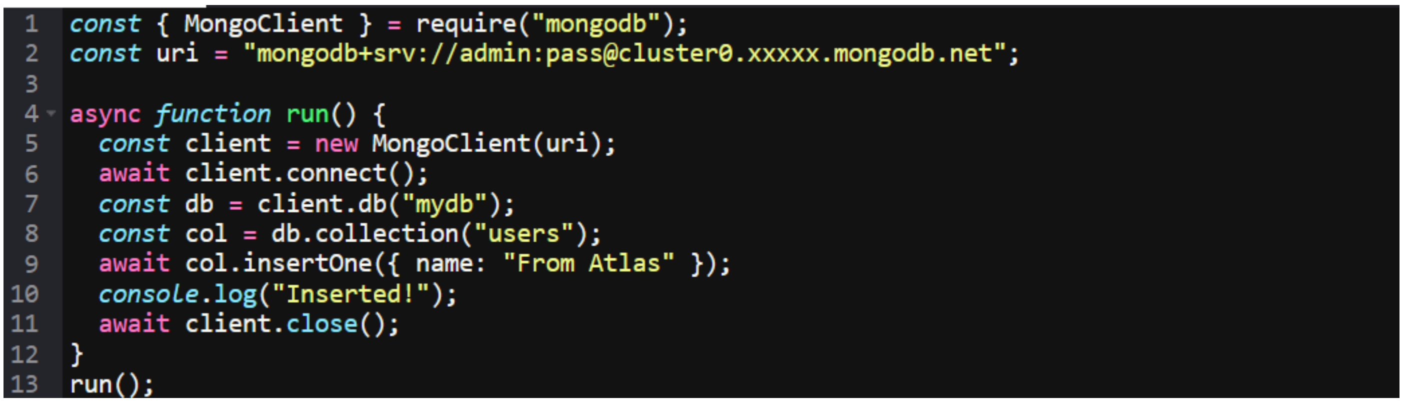
MongoDB Atlas (Cloud) – Free Forever

1. Go to: https://cloud.mongodb.com
2. Sign up (Google/GitHub)
3. Create **Cluster** → **M0 Free**
4. Whitelist IP: 0.0.0.0/0 (for learning)
5. Create user: admin / password123
6. Get connection string:

mongodb+srv://admin:password123@cluster0.xxxxx.mongodb.net/

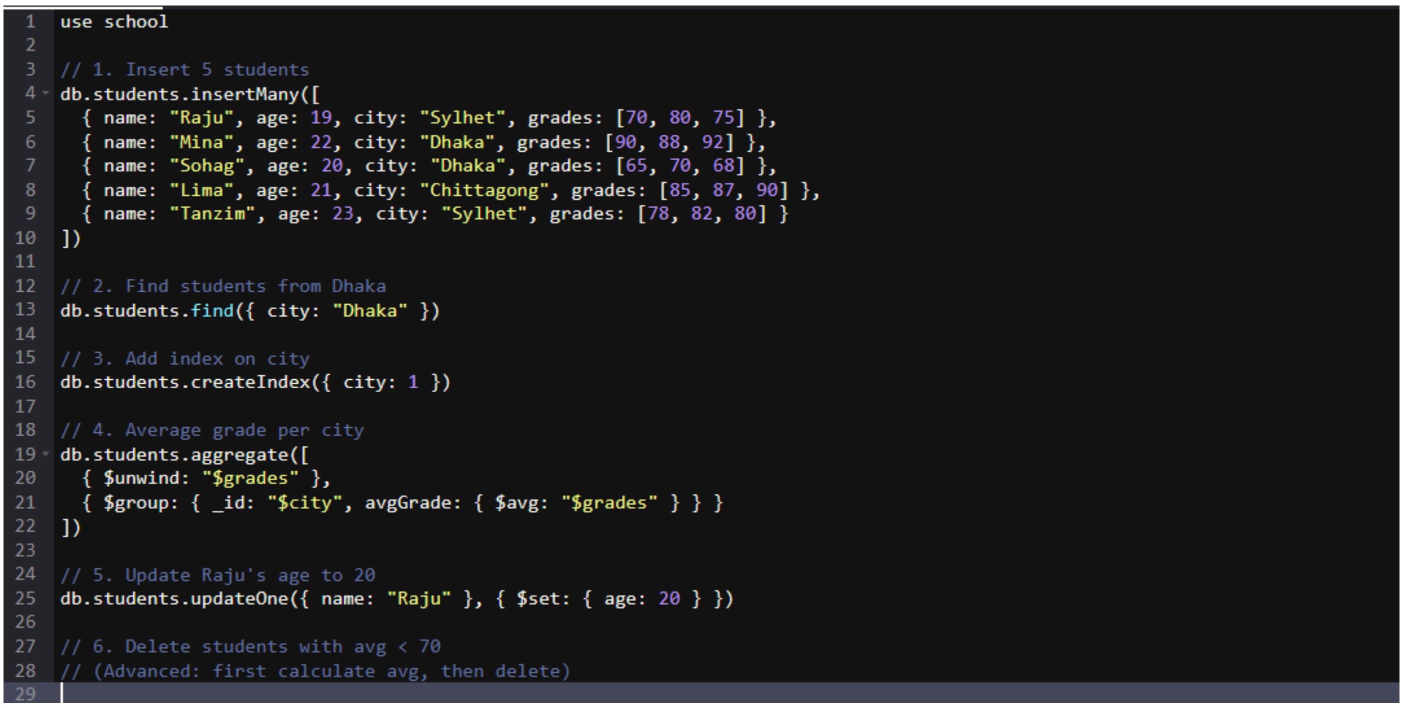
7. Use in Compass or mongosh

Use in Code (Node.js Example)



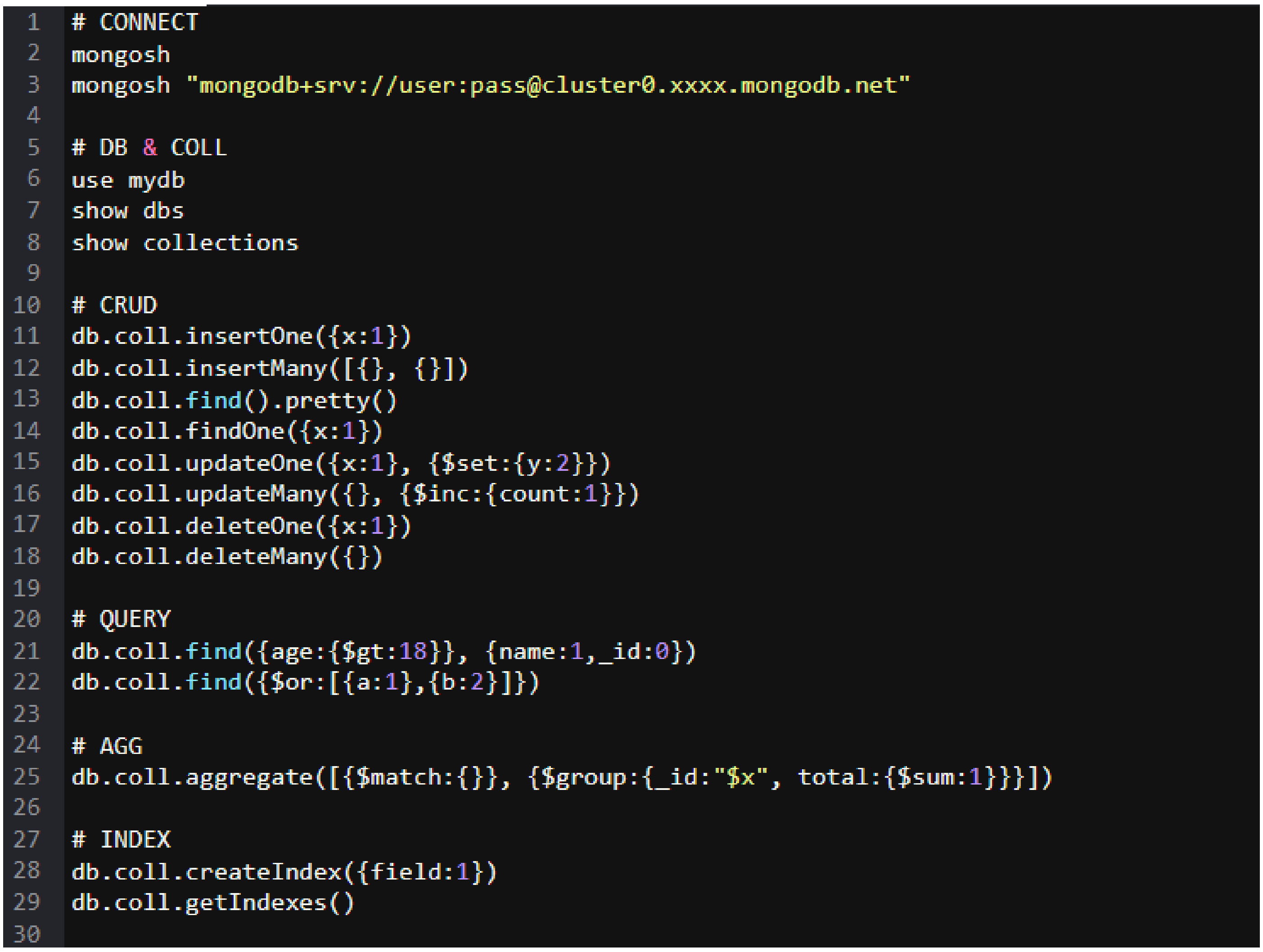
```
const { MongoClient } = require("mongodb");
const uri = "mongodb+srv://admin:pass@cluster0.xxxxx.mongodb.net";

async function run() {
  const client = new MongoClient(uri);
  await client.connect();
  const db = client.db("mydb");
  const col = db.collection("users");
  await col.insertOne({ name: "From Atlas" });
  console.log("Inserted!");
  await client.close();
}
run();
```

Mini Project (Practice!)



```
use school

// 1. Insert 5 students
db.students.insertMany([
  { name: "Raju", age: 19, city: "Sylhet", grades: [70, 80, 75] },
  { name: "Mina", age: 22, city: "Dhaka", grades: [90, 88, 92] },
  { name: "Sohag", age: 20, city: "Dhaka", grades: [65, 70, 68] },
  { name: "Lima", age: 21, city: "Chittagong", grades: [85, 87, 90] },
  { name: "Tanzim", age: 23, city: "Sylhet", grades: [78, 82, 80] }
])

// 2. Find students from Dhaka
db.students.find({ city: "Dhaka" })

// 3. Add index on city
db.students.createIndex({ city: 1 })

// 4. Average grade per city
db.students.aggregate([
  { $unwind: "$grades" },
  { $group: { _id: "$city", avgGrade: { $avg: "$grades" } } }
])

// 5. Update Raju's age to 20
db.students.updateOne({ name: "Raju" }, { $set: { age: 20 } })

// 6. Delete students with avg < 70
// (Advanced: first calculate avg, then delete)
```

Cheat Sheet (Print & Stick!)



```
# CONNECT
mongosh
mongosh "mongodb+srv://user:pass@cluster0.xxxx.mongodb.net"

# DB & COLL
use mydb
show dbs
show collections

# CRUD
db.coll.insertOne({x:1})
db.coll.insertMany([{}, {}])
db.coll.find().pretty()
db.coll.findOne({x:1})
db.coll.updateOne({x:1}, {$set:{y:2}})
db.coll.updateMany({}, {$inc:{count:1}})
db.coll.deleteOne({x:1})
db.coll.deleteMany({})

# QUERY
db.coll.find({age:{$gt:18}}, {name:1,_id:0})
db.coll.find({$or:[{a:1},{b:2}]})

# AGG
db.coll.aggregate([{$match:{}}, {$group:{_id:"$x", total:{$sum:1}}}])

# INDEX
db.coll.createIndex({field:1})
db.coll.getIndexes()
```