Database Overview

Database Name: granular_test
Host: mongodb://localhost:27017/

Users Collection Structure

Field	Type	Description
_id	ObjectId	Primary Key (Auto-generated)
name	String	User's name
salary	Number	User's salary (sensitive)
department	String	User's department

Sample Data:

```
use granular_test

db.users.insertMany([
  { name: "Alice", salary: 50000, department: "HR" },
  { name: "Bob", salary: 60000, department: "Engineering" },
  { name: "Charlie", salary: 70000, department: "Finance" }
])
```

Connection Details

Prerequisites

- Install Docker to run the MongoDB container locally.

Fetch MongoDB image and run the container:
```
docker run --name mongodb-container -e MONGO_INITDB_ROOT_USERNAME=root
-e MONGO_INITDB_ROOT_PASSWORD=examplepassword -p 27017:27017 -d mongo
```

Database Credentials

Role	Username	Password	Access Rights
Read-Only	readonly_user	readpass	Read-only (id, name)
Read-Write	readwrite_user	writepass	Full access to users collection
Integrator	integrator	integratorpass	Can read only id, department (Engineering only)

MongoDB Connection Strings:

mongodb://readonly_user:readpass@localhost:27017/granular_test
mongodb://readwrite_user:writepass@localhost:27017/granular_test
mongodb://integrator:integratorpass@localhost:27017/granular_test

---

Setting Up the Database

Step 1: Access the MongoDB CLI

docker exec -it mongodb-container mongosh -u root -p examplepassword --authenticationDatabase admin

Step 2: Create Roles and Users

change db: use granular_test

1. Create Read-Only Role

```
db.createRole({
  role: "readOnlyUsers",
  privileges: [{ resource: { db: "granular_test", collection: "users" }, actions: ["find"] }],
  roles: []
})

db.createUser({
  user: "readonly_user",
  pwd: "readpass",
  roles: [{ role: "readOnlyUsers", db: "granular_test" }]
})
```

2. Create Read-Write Role

```
db.createRole({
  role: "readWriteUsers",
```

```
  privileges: [{ resource: { db: "granular_test", collection: "users" }, actions: ["find", "insert", "update", "remove"] }],
  roles: []
})

db.createUser({
  user: "readwrite_user",
  pwd: "writepass",
  roles: [{ role: "readWriteUsers", db: "granular_test" }]
})
```

3. Create Integrator Role (Engineering Only Access)

```
db.createRole({
  role: "integratorRole",
  privileges: [{ resource: { db: "granular_test", collection: "users" }, actions: ["find"] }],
  roles: []
})

db.createUser({
  user: "integrator",
  pwd: "integratorpass",
  roles: [{ role: "integratorRole", db: "granular_test" }]
})
```

Testing Granular Access

Logging In

when loggin in, you should change the database to “granular_test” like this:
use granular_test

To test access for each role, log in using the appropriate credentials:

```
docker exec -it mongodb-container mongosh -u readonly_user -p readpass --authenticationDatabase granular_test
docker exec -it mongodb-container mongosh -u readwrite_user -p writepass --authenticationDatabase granular_test
docker exec -it mongodb-container mongosh -u integrator -p integratorpass --authenticationDatabase granular_test
```

Role Access Tests

Read-Only User (`readonly_user`)

Test 1:

db.users.find()

Expected Result: Only `id` and `name` fields visible.

Test 2:

db.users.insertOne({ name: "Dave", salary: 80000, department: "Sales" })

Expected Error: **Unauthorized**

Read-Write User (`readwrite_user`)

Test 1:

db.users.find()

Expected Result: All fields visible.

Test 2:

db.users.updateOne({ name: "Alice" }, { $set: { salary: 55000 } })

Expected Result: Salary updated successfully.

Integrator (`integrator`)

Test 1:

db.users.find()

Expected Result: Only `id` and `department` fields for **Engineering** department rows visible.

Test 2:

db.users.find({}, { name: 1 })

Expected Error: **Unauthorized access to `name` field**

Test 3:

```
db.users.insertOne({ name: "Eve", salary: 90000, department: "Finance" })
```

Expected Error: **Unauthorized**

Summary of Granular Access

User	Cannot Read	Can Only Read	Can Read & Write
readonly_user	`salary`, `department`	`id`, `name`	None
readwrite_user	None	`id`, `name`, `salary`, `department`	All columns
integrator	`name`, `salary`, non-Engineering rows	`id`, `department` (Engineering only)	None

---