**docker exec mongodb mongodump --host=localhost --port=27017 -u root -p rootpassword --authenticationDatabase=admin --out=/data/mongodump/**

**root@ip-172-31-3-127:~# docker exec mongodb mongodump --host=localhost --port=27017 -u root -p rootpassword --authenticationDatabase=admin --out=/data/mongodump/**

**2025-03-03T10:34:26.260+0000 writing admin.system.users to /data/mongodump/admin/system.users.bson**

**2025-03-03T10:34:26.261+0000 done dumping admin.system.users (1 document)**

**2025-03-03T10:34:26.262+0000 writing admin.system.version to /data/mongodump/admin/system.version.bson**

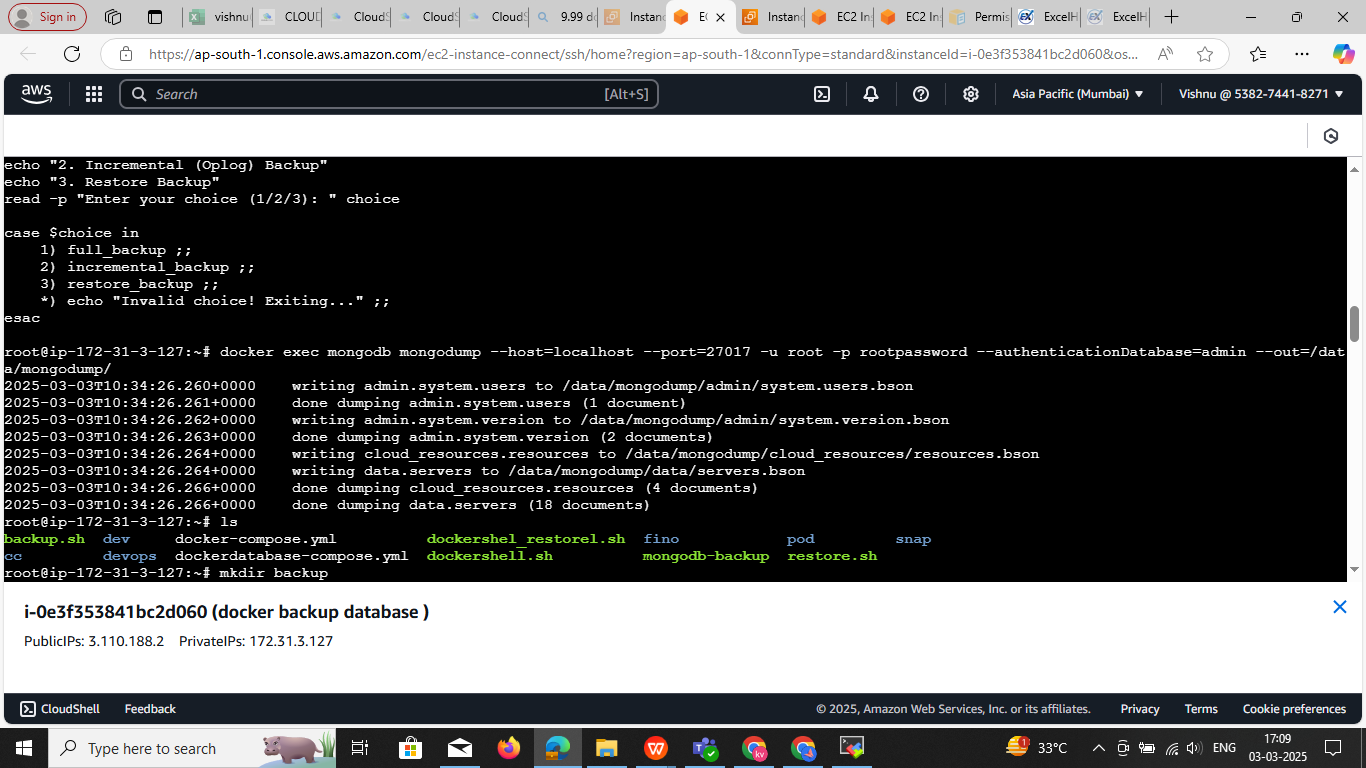
**2025-03-03T10:34:26.263+0000 done dumping admin.system.version (2 documents)**

**2025-03-03T10:34:26.264+0000 writing cloud\_resources.resources to /data/mongodump/cloud\_resources/resources.bson**

**2025-03-03T10:34:26.264+0000 writing data.servers to /data/mongodump/data/servers.bson**

**2025-03-03T10:34:26.266+0000 done dumping cloud\_resources.resources (4 documents)**

**2025-03-03T10:34:26.266+0000 done dumping data.servers (18 documents)**



mkdir backup got to dir check the dir any files

**root@ip-172-31-3-127:~/backup# ls -la**

**total 8**

**drwxr-xr-x 2 root root 4096 Mar 3 10:34 .**

**drwx------ 11 root root 4096 Mar 3 10:34 ..**

docker cp mongodb:/data/mongodump **./backup**

**cd backup/**

**root@ip-172-31-3-127:~/backup# ls -la**

**total 12**

**drwxr-xr-x 3 root root 4096 Mar 3 10:36 .**

**drwx------ 11 root root 4096 Mar 3 10:34 ..**

**drwxr-xr-x 6 root root 4096 Feb 28 15:36 mongodump**

**root@ip-172-31-3-127:~/backup/mongodump#** ls -la ckeck the data

**total 24**

**drwxr-xr-x 6 root root 4096 Feb 28 15:36 .**

**drwxr-xr-x 3 root root 4096 Mar 3 10:36 ..**

**drwxr-xr-x 2 root root 4096 Feb 28 15:17 admin**

**drwxr-xr-x 2 root root 4096 Feb 28 15:17 cloud\_resources**

**drwxr-xr-x 2 root root 4096 Feb 28 15:17 data**

**drwxr-xr-x 5 root root 4096 Feb 28 15:17 mongodump**

**docker exec mongodb mongorestore --username root --password rootpassword --authenticationDatabase admin --drop /data/mongodump**

· If you restore with --drop, **MongoDB erases existing collections and restores everything** from the backup.

· To restore **only missing documents**, remove --drop.

· To restore **only one document**, extract it manually and insert it into MongoDB.

**With out drop command :**

docker exec mongodb mongorestore --username root --password rootpassword --authenticationDatabase admin --db mydatabase --collection resources /data/mongodump/mydatabase/resources.bson

This command **restores a specific collection (**resources**) from a backup into a MongoDB database (**mydatabase**)** inside a Docker container.

### ****Breakdown of the Command****

| **Part** | **Meaning** |
| --- | --- |
| docker exec mongodb | Runs a command inside the **MongoDB Docker container** named mongodb. |
| mongorestore | The **MongoDB restore tool** used to import data from a backup (.bson file). |
| --username root | Specifies the **MongoDB admin username** (root). |
| --password rootpassword | Specifies the **MongoDB admin password** (rootpassword). |
| --authenticationDatabase admin | Authenticates against the admin database (required for privileged users). |
| --db mydatabase | Restores data into the mydatabase database. |
| --collection resources | Restores only the resources collection. |
| /data/mongodump/mydatabase/resources.bson | The **path to the** .bson **backup file** inside the container. |

**2025-03-03T11:10:59.359+0000 preparing collections to restore from**

**2025-03-03T11:10:59.360+0000 don't know what to do with subdirectory "mongodump/admin", skipping...**

**2025-03-03T11:10:59.360+0000 don't know what to do with subdirectory "mongodump/cloud\_resources", skipping...**

**2025-03-03T11:10:59.360+0000 don't know what to do with subdirectory "mongodump/data", skipping...**

**2025-03-03T11:10:59.362+0000 reading metadata for data.servers from /data/mongodump/data/servers.metadata.json**

**2025-03-03T11:10:59.363+0000 reading metadata for cloud\_resources.resources from /data/mongodump/cloud\_resources/resources.metadata.json**

**2025-03-03T11:10:59.364+0000 dropping collection data.servers before restoring**

**2025-03-03T11:10:59.367+0000 dropping collection cloud\_resources.resources before restoring**

**2025-03-03T11:10:59.402+0000 restoring cloud\_resources.resources from /data/mongodump/cloud\_resources/resources.bson**

**2025-03-03T11:10:59.409+0000 restoring data.servers from /data/mongodump/data/servers.bson**

**2025-03-03T11:10:59.418+0000 finished restoring cloud\_resources.resources (4 documents, 0 failures)**

**2025-03-03T11:10:59.421+0000 finished restoring data.servers (18 documents, 0 failures)**

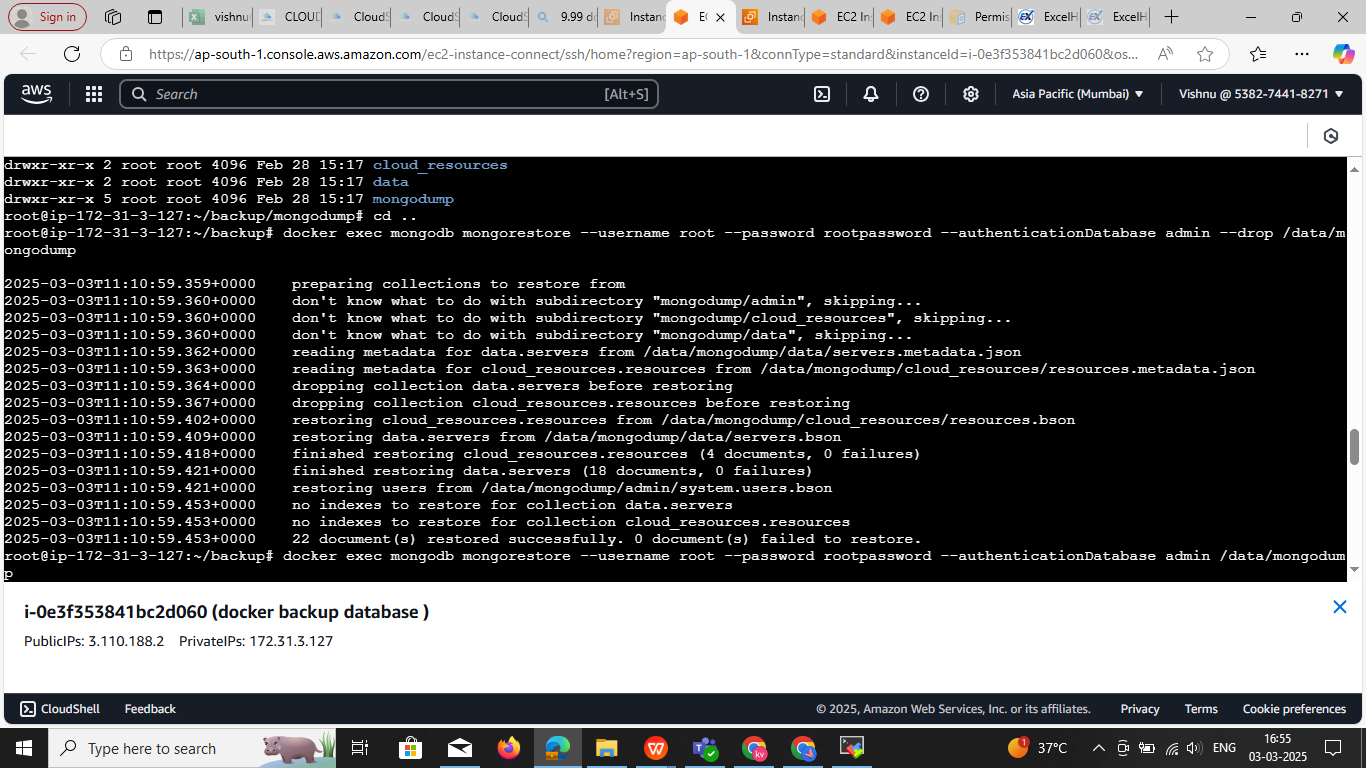
**2025-03-03T11:10:59.421+0000 restoring users from /data/mongodump/admin/system.users.bson**

**2025-03-03T11:10:59.453+0000 no indexes to restore for collection data.servers**

**2025-03-03T11:10:59.453+0000 no indexes to restore for collection cloud\_resources.resources**

**2025-03-03T11:10:59.453+0000 22 document(s) restored successfully. 0 document(s) failed to restore.**

**Screen :**



docker exec mongodb mongorestore --username root --password rootpassword --authenticationDatabase admin /data/mongodump

### ****What Happens When You Run This Command?****

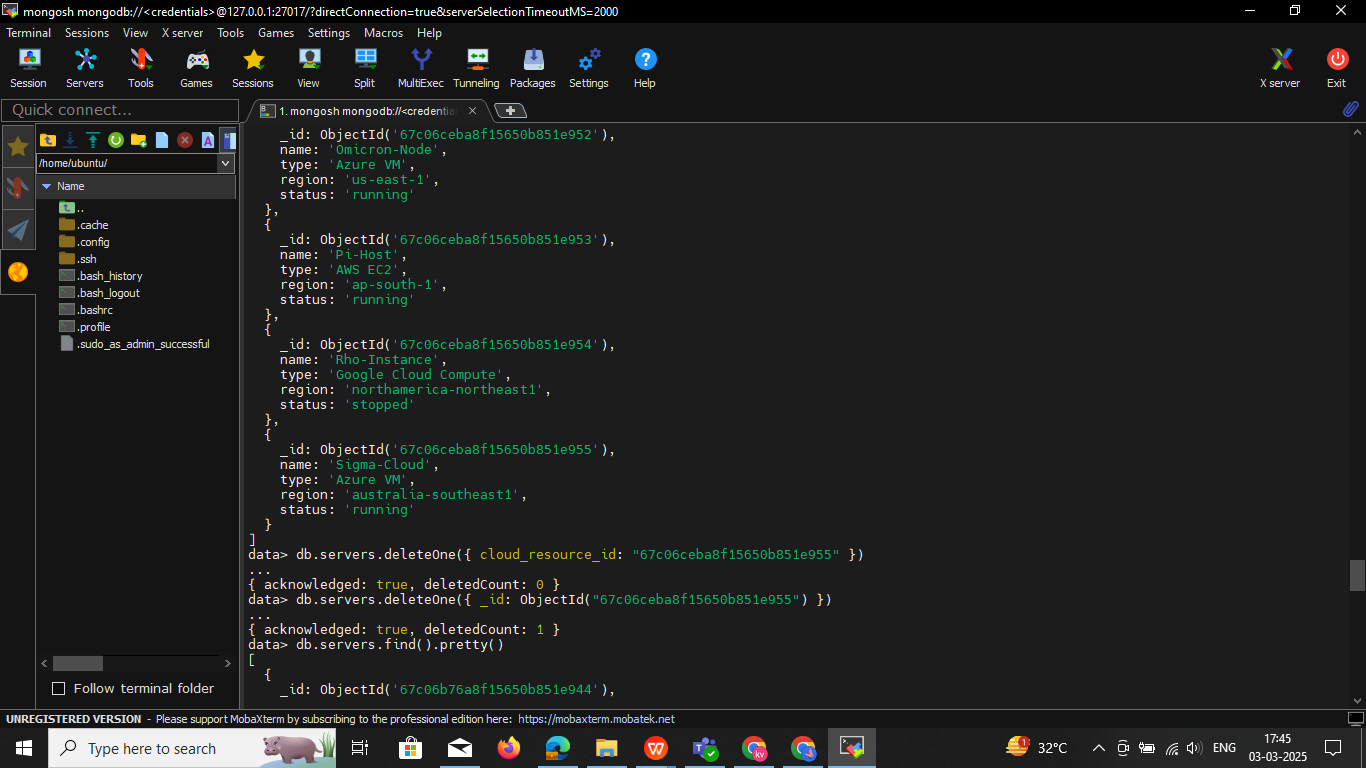
1. The mongodb **container** runs the mongorestore tool.
2. **Authentication** occurs using root/rootpassword in the admin database.
3. **Only the** resources **collection** from /data/mongodump/mydatabase/resources.bson is restored into the mydatabase database.
4. If the resources collection **already exists**, documents from the backup are inserted **without deleting existing ones**.
5. If you want to **replace existing data**, use the --drop option:

**All commands 5 commands dump and restore**

1 **docker exec mongodb mongodump --host=localhost --port=27017 --username=root --password=rootpassword --authenticationDatabase=admin --out=/data/mongodump/**

2 **docker cp mongodb:/data/mongodump ./cc** cc dir

3 cd ~/cc go --> mongo db login --> drope db



4 **docker exec mongodb mongorestore --host=localhost --port=27017 -u root -p rootpassword --authenticationDatabase=admin --dir=/data/mongodump --drop**

Using the --drop flag in the mongorestore command ensures that existing collections are **deleted before restoring** new data. This is useful when you want to **avoid duplicate records** or **ensure a fresh start** with the restored backup.

### **Key Benefits of --drop:**

1. **Prevents Data Duplication** – Drops existing collections before restoring, avoiding conflicts.
2. **Ensures a Clean Restore** – Old data is removed, making sure only the backup data exists.
3. **Avoids Schema Conflicts** – If the structure of collections has changed, --drop prevents errors by removing old ones first.
4. **Useful for Testing & Development** – Helps reset databases without manually deleting collections.

**Caution:** The --drop flag **permanently deletes existing collections before restoring**. Use it carefully to avoid accidental data loss.

5 **docker exec mongodb mongorestore --username root --password rootpassword --authenticationDatabase admin --db mydatabase --collection resources /data/mongodump/mydatabase/resources.bson**

./mongodb-backup.sh full

./mongodb-backup.sh incr

./mongodb-backup.sh restore

./mongodb-backup.sh cleanup