Day3_1_Self-Managed Backup & Recovery

- Logical backups (mongodump/mongorestore)
- Physical backups (Filesystem snapshot)
- Hot vs cold backup
- PITR (Point-in-Time Recovery)
- Lab: Perform backup & restore . Create Detail Hands on Examples Step by

This is structured exactly like a training lab for **MongoDB Administrators**, with:

- Scenario & Use Case
- Logical backups
- Physical backups (filesystem snapshots)
- Hot vs Cold backup methods
- PITR (Point-in-Time Recovery)
- Hands-on terminal commands
- Validation steps for DBAs

⭐ CSC Scenario — Real-World Use Case

- CSC runs an internal **Compliance & Filings** system for handling millions of corporate filings.

- Bangalore manages the **primary MongoDB deployment** (self-managed, on-prem or VM).
- Mumbai needs **regular backups**, **hot backups**, and **test environment restores**.

- During US year-end filing season, downtime is unacceptable—so backups must be **safe**, **consistent**, and **recoverable**.

Your job as a DBA:

1. Create **logical backups** using mongodump.
2. Create **physical backups** using filesystem snapshots.
3. Understand **hot vs cold** backups.
4. Perform **point-in-time recovery (PITR)** using oplog.
5. Fully restore a corrupted collection/database.

🌐 Environment Assumptions

- MongoDB installed at:

  /var/lib/mongo (Linux)

- Config file at:

  /etc/mongod.conf

- Data directory:

  /var/lib/mongo

- Backup folder created at:

  /backups/csc

- Database name:

  csc_compliance

- Collections:

  entities, filings, reminders

- Oplog enabled (Replica Set)

1️⃣ **Logical Backups —**

mongodump/ mongorestore

Logical backup = BSON dump of data (**not** data files).

✔ Good for portability

✔ Good for schema migrations

✔ Good for restoring single collection

❌ Slow for huge datasets (> 500GB)

Step 1. Create a Directory for Backups

```
sudo mkdir -p /backups/csc
sudo chown $(whoami) /backups/csc
```

Step 2. Take Full Backup with mongodump

Command:

```
mongodump \
  --host localhost \
  --port 27017 \
  --db csc_compliance \
  --out /backups/csc/backup_full_$(date +%F)
```

Output folder structure:

```
/backups/csc/backup_full_2025-01-01/
    csc_compliance/
        entities.bson
        filings.bson
        reminders.bson
        metadata.json
```

✔ **CSC Use Case:**

Bangalore DBA takes a daily backup before deployment.

Step 3. Take Backup of a Single Collection

```
mongodump \
  --db csc_compliance \
  --collection filings \
  --out /backups/csc/filings_only_$(date +%F)
```

✔ **CSC Use Case:**

A new patch modifies only the **filings** schema.
Mumbai requests a ***collection-only*** backup before applying the script.

Step 4. Restore Full Database with

mongorestore

```
mongorestore \
  --db csc_compliance_test \
  /backups/csc/backup_full_2025-01-01/csc_compliance
```

Now check:

```
mongosh
use csc_compliance_test
show collections
db.filings.countDocuments()
```

✓ **CSC Use Case:**

Mumbai QA team gets a **restored test copy** for UAT.

Step 5. Restore a Single Collection

```
mongorestore \
  --db csc_compliance \
  --collection filings \
  /backups/csc/filings_only_2025-01-
01/csc_compliance/filings.bson
```

2️⃣ **Physical Backups — Filesystem Snapshot (LVM / EBS / VMware)**

Physical backup = copying **raw data files**.

✓ Very fast

✓ Perfect for large datasets

❌ Must ensure consistency

❌ Requires filesystem-level tools

Step 1. Stop Writes (if doing COLD backup)

Cold backup means **mongod stopped**.

```
sudo systemctl stop mongod
```

Copy the files:

```
sudo cp -R /var/lib/mongo /backups/csc/mongo_cold_2025_01_01
```

Restart:

```
sudo systemctl start mongod
```

✓ **CSC Use Case:**

During planned maintenance window, Bangalore DBA takes a **cold physical backup**.

Step 2. HOT Physical Snapshot (Preferred for Production)

Hot backups = system **keeps running**.

Requirements:

- Replica set (primary + secondary)
- Run snapshot on **secondary**
- Use LVM / ZFS / AWS EBS snapshot

Step-by-step:

1️⃣ Run backup on SECONDARY node

Check replica status:

```
mongosh
rs.status()
```

Identify **secondary** server.

2️⃣ Freeze the filesystem

```
fsfreeze -f /var/lib/mongo
```

3️⃣ Take snapshot (example: LVM)

```
lvcreate -L 20G -s -n mongo_snapshot /dev/vg0/mongodata
```

4️⃣ Unfreeze filesystem

```
fsfreeze -u /var/lib/mongo
```

5️⃣ Mount snapshot & copy back:

```
mount /dev/vg0/mongo_snapshot /mnt/snapshot
cp -R /mnt/snapshot /backups/csc/hot_snapshot_2025_01_01
umount /mnt/snapshot
```

✔ CSC Use Case:

Bangalore SRE team uses **hot LVM snapshots** on secondary nodes to take backups without downtime during heavy filing periods.

3️⃣ Hot vs Cold Backup — Simple Comparison

Feature	Hot Backup	Cold Backup
Database Running	✓ Yes	❌ No
Downtime	None	Required
Risk	Low (on Secondary)	Lowest
Difficulty	Medium	Easy
Used for CSC	24×7 workloads	Maintenance windows

4️⃣ PITR — Point-in-Time Recovery (Using Oplog)

PITR allows restoring data **up to a certain timestamp**.

Requirements:

- Replica set (oplog enabled)
- Continuous oplog backups

Step 1. Take Full Logical Backup

```
mongodump \
  --oplog \
  --out /backups/csc/full_with_oplog_2025_01_01
```

This folder now contains:

```
csc_compliance/
oplog.bson
```

Step 2. System Fails — Example: Accidental Delete

Mumbai developer accidentally runs:

```
db.filings.deleteMany({ state: "DE" });
```

Step 3. Restore Up to a Safe Timestamp

Suppose correct timestamp is:

```
2025-01-01T10:30:00Z
```

Restore:

```
mongorestore \
  --oplogReplay \
  --oplogLimit "2025-01-01T10:30:00Z" \
  /backups/csc/full_with_oplog_2025_01_01
```

✔ **CSC Use Case:**

Developer accidentally deleted **all Delaware filings.**
Bangalore DBA restores database exactly to **10:30 AM**, 3 minutes before deletion.

5️⃣ Full Hands-On LAB (Step-by-Step for CSC Teams)

Use this in a 90-minute workshop.

🔥

LAB 1 — Take a Logical Backup

Step A: Run backup command

```
mongodump --db csc_compliance --out /backups/csc/lab_backup
```

Step B: Verify contents

```
ls /backups/csc/lab_backup/csc_compliance/
```

🔥

LAB 2 — Corrupt the Data (Controlled)

Switch DB:

```
mongosh
use csc_compliance
db.filings.deleteMany({})
```

Verify:

```
db.filings.countDocuments()   // 0
```

🔥

LAB 3 — Restore Using mongorestore

```
mongorestore --db csc_compliance
/backups/csc/lab_backup/csc_compliance
```

Check:

```
db.filings.countDocuments()
```

🔥

LAB 4 — Create a Hot Snapshot (Simulated)

If using Linux VM:

```
sudo fsfreeze -f /var/lib/mongo
# take snapshot (LVM/EBS/vSphere)
sudo fsfreeze -u /var/lib/mongo
```

LAB 5 — PITR Simulation

Step A: Take backup with oplog

```
mongodump --oplog --out /backups/csc/oplog_backup
```

Step B: Make a bad write

```
db.entities.updateMany({}, { $set: { status: "Inactive" } })
```

Step C: Restore before corruption

```
mongorestore --oplogReplay --oplogLimit "2025-01-01T10:05:00Z"
/backups/csc/oplog_backup
```

⭐ **Summary for CSC DBAs**

Backup Type	When CSC Should Use It
Logical	Small DBs, migrations, restoring specific collections
Physical	Large production DBs; fast snapshots
Hot Backup	Zero downtime requirement
Cold Backup	Maintenance windows

Backup Type	**When CSC Should Use It**
PITR	Recover from accidental delete, logic bugs

Below is a **combined Hands-On Guide for Self-Managed Backup & Recovery** for **Windows**, **Mac**, and **MongoDB Atlas Clusters**, tailored for **CSC (Corporation Service Company)** teams in **Bangalore & Mumbai.**

This guide expands the earlier Linux-based tutorial into **true multi-platform coverage**, showing how DBAs work across:

- On-prem Windows servers
- Mac developer machines (for testing restores)
- Cloud environments (MongoDB Atlas)

⭐ CSC BUSINESS USE CASE (Tell this story to participants)

CSC runs a large **Compliance & Filings Platform.**

- **Bangalore** team manages **production & DR backup plans.**
- **Mumbai** team needs **daily logical backups**, **snapshot backups**, and **Atlas backups** for testing, recovery drills, and UAT.

They operate with:

- Windows servers (legacy systems)
- Mac laptops (developer machines)
- MongoDB Atlas clusters (production cloud)

You must learn how to:
✓ Perform **logical backups** (portable)
✓ Perform **physical backups** (fast)
✓ Do **hot & cold backups**
✓ Perform **PITR** (Point-in-Time Recovery)
✓ Backup & restore on **Windows**, **Mac**, and **Atlas**

This is a **cross-platform MongoDB admin capability workshop**.

🔧 IMPORTANT — TOOL INSTALLATION (Windows & Mac)

➤ Install MongoDB Database Tools

(This includes mongodump, mongorestore.)

Download here from MongoDB website:

Windows:

```
https://www.mongodb.com/try/download/database-tools
```

After installation, ensure PATH includes:

```
C:\Program Files\MongoDB\Tools\bin
```

Mac (Homebrew):

```
brew tap mongodb/brew
brew install mongodb-database-tools
```

🔧 PART 1 — LOGICAL BACKUPS (mongodump / mongorestore)

Works the SAME on Windows, Mac & Atlas (only connection string differs).

Logical backup = BSON export files.

✔ Best for portability

✔ Good for dev/test

✔ Allows single collection restore

❌ Slower for huge data

1️⃣ WINDOWS — Logical Backup & Restore

✔ Step 1: Open Command Prompt

```
mongodump --db=csc_compliance --
out="C:\backups\csc\backup_2025_01_01"
```

✔ Step 2: Restore

```
mongorestore --db=csc_compliance_test
"C:\backups\csc\backup_2025_01_01\csc_compliance"
```

✔ Step 3: Single Collection Backup

```
mongodump --db=csc_compliance --collection=filings --
out="C:\backups\csc\filings"
```

✔ **CSC Use Case:**

Bangalore Windows server running MongoDB needs nightly backups and collection-level rollback files.

2️⃣ MAC — Logical Backup & Restore

✔ Step 1: Backup

```
mongodump --db=csc_compliance --out
~/csc_backups/full_2025_01_01
```

✔ Step 2: Restore

```
mongorestore --db=csc_restore_test
~/csc_backups/full_2025_01_01/csc_compliance
```

✔ Step 3: Backup One Collection

```
mongodump --db=csc_compliance --collection=entities --out
~/csc_backups/entities_backup
```

✔ **CSC Use Case:**

Mumbai developers take logical dumps on MacBooks to reproduce production bugs locally.

3️⃣ ATLAS — Logical Backup & Restore

You must use the **MongoDB Atlas connection string.**

✔ Step 1: Backup entire DB

```
mongodump \
  --
uri="mongodb+srv://csc_app_user:Password@cluster0.abc.mongodb.
net/csc_compliance" \
  --out=./atlas_backup_2025_01_01
```

✔ Step 2: Restore into testing DB (Atlas or local)

```
mongorestore \
  --nsFrom="csc_compliance.*" \
  --nsTo="csc_compliance_test.*" \
  ./atlas_backup_2025_01_01
```

✔ **CSC Use Case:**

Mumbai QA asks Bangalore team for a sanitized subset of production data to test a new filings workflow.

🔧 PART 2 — PHYSICAL BACKUPS (File Copy / Snapshots)

Physical backup = copying MongoDB **data files on disk**.

✔ Fast

✔ Great for huge DB sizes

❌ Must handle lock consistency

❌ Only for self-managed (not Atlas)

1️⃣ WINDOWS — Physical Backup

Data folder location (default):

```
C:\Program Files\MongoDB\Server\6.0\data
```

Cold Backup (mongod stopped)

1. Stop MongoDB:

```
net stop MongoDB
```

2. Copy folder:

```
xcopy "C:\Program Files\MongoDB\Server\6.0\data"
"D:\csc_backups\data_2025_01_01" /E /I /H
```

3. Start MongoDB:

```
net start MongoDB
```

✔ **Use Case:**

During scheduled maintenance window, Bangalore DBA takes a cold snapshot of the legacy Windows MongoDB server.

Hot Backup (VSS Snapshot)

MongoDB does NOT natively handle hot physical backup on Windows, but **Windows Volume Shadow Copy Service (VSS)** works.

1. Use disk-level snapshot tool (Veeam, Acronis, Windows Server Backup).
2. Snapshot is taken **without stopping mongod.**

✔ **Use Case:**

CSC infrastructure team uses VSS-backed snapshots every hour with zero downtime.

2️⃣ MAC — Physical Backup

Mac is usually used for *development*, but physical backup still works.

Cold Backup:

```
brew services stop mongodb-community
```

```
cp -R /usr/local/var/mongodb ~/csc_backup_2025_01_01
brew services start mongodb-community
```

✔ Use Case:
Mac developer wants a safe snapshot before running schema migration scripts.

3️⃣ ATLAS — Physical Backup

❌ Not allowed.
MongoDB Atlas does **not** allow direct access to DB files.

Instead, Atlas provides:

✔ Snapshots (Fully automated, point-in-time)

✔ Continuous Backups for PITR

🔧 PART 3 — HOT vs COLD BACKUP (Explained Simply)

Mode	DB Running?	Downtime	Use Case
Hot Backup	✔ Yes	No	CSC production servers, Atlas
Cold Backup	❌ No	Yes	Windows VM maintenance window
Snapshot (Hot)	✔ Yes	No	Linux LVM, Windows VSS

🔧 PART 4 — Point-in-Time Recovery (PITR)

PITR lets CSC recover data to **any moment** (before delete, mistake, corruption).

1️⃣ WINDOWS – PITR using Oplog

Only works if MongoDB is **Replica Set**.

Step 1: Backup with oplog

```
mongodump --db=csc_compliance --out="C:\csc_backup" --oplog
```

Files created:

```
csc_compliance/
oplog.bson
```

Step 2: Developer accidentally deletes:

```
db.filings.deleteMany({ state: "DE" })
```

Step 3: Restore up to time before deletion

```
mongorestore --oplogReplay --oplogLimit "2025-01-01T12:30:00Z"
C:\csc_backup
```

✓ CSC Use Case:
Developer in Mumbai accidentally wipes **Delaware filings**; Bangalore restores them to **12:30 PM** safely.

2️⃣ MAC — PITR

Commands same as Windows:

```
mongodump --oplog --out ~/csc_pitr

mongorestore --oplogReplay --oplogLimit "2025-01-01T10:15:00Z"
~/csc_pitr
```

3️⃣ ATLAS — PITR

Atlas supports PITR only on:

- **Dedicated clusters (M10+)**
- **Continuous Backup enabled**

Steps:

1. Go to **Atlas → Backups**
2. Select **Continuous Backup**
3. Choose **Point in Time**
4. Select timestamp
5. Restore to:
    - **Same cluster**
    - **New cluster** (recommended for safety)

✔ CSC Use Case:

During filings peak, an engineer deletes 200k filings due to wrong query.

Atlas PITR lets Bangalore restore precisely to **10:24:52 AM IST**.

PART 5 — FULL HANDS-ON LAB (Windows + Mac + Atlas)

Use this as a training workflow for CSC admin teams.

LAB STEP 1 — Create Logical Backup (Win/Mac/Atlas)

Windows:

```
mongodump --db=csc_compliance --out C:\csc\backup1
```
Mac:
```
mongodump --db=csc_compliance --out ~/csc_backup1
```
Atlas:
```
mongodump --uri "mongodb+srv://..." --out ./atlas_backup1
```

LAB STEP 2 — Verify Backup Files

List folder contents.

LAB STEP 3 — Simulate Data Loss

```
db.reminders.deleteMany({})
```

LAB STEP 4 — Restore Backup

Windows:

```
mongorestore C:\csc\backup1\csc_compliance
```
Mac:
```
mongorestore ~/csc_backup1/csc_compliance
```
Atlas → Local Test:
```
mongorestore ./atlas_backup1
```

LAB STEP 5 — PITR Restoration

Take backup with oplog, delete data, restore with timestamp.

LAB STEP 6 — Snapshot Backup (Optional)

Windows VSS or Mac copy-based.

LAB STEP 7 — Validate DB

Check counts, sample docs.

⭐ SUMMARY FOR CSC ADMINS

Topic	Windows	Mac	Atlas
Logical Backup	✓ mongodump	✓ mongodump	✓ via URI
Physical Backup	✓ Copy data folder, VSS	✓ Copy folder	❌ Not allowed
PITR	✓ oplog	✓ oplog	✓ Atlas Continuous
Hot Backup	✓ VSS	✓ Mac VM snapshot	✓ Built-in
Cold Backup	✓ Stop service	✓ Stop service	❌