🔄 Follow-Up Task: Restore the Database from Your Backup Files

🏥 Scenario: Emergency Recovery Simulation for TrainingDB

Today, you practiced backing up your database TrainingDB using different types of backups:

✅ Full Backup

✅ Differential Backup

✅ Transaction Log Backup

✅ Copy-Only Backup

Now simulate this real-world problem:

⚠️ This afternoon, the TrainingDB system crashed. Your manager asks you to **recover the database up to the last transaction log backup** you made. Your goal is to bring the database back to the most recent state — using your backup files only.

Instructions:

Step 1: Drop the Current Database (Simulate System Failure)

```
DROP DATABASE TrainingDB;
```

Step 2: Restore from Your Backups

Use the **same file names and paths** you used earlier. Replace them accordingly.

```
-- 1. Restore FULL backup

RESTORE DATABASE TrainingDB

FROM DISK = 'C:\Backups\TrainingDB_Full.bak'

WITH NORECOVERY;


-- 2. Restore DIFFERENTIAL backup (if you created one)

RESTORE DATABASE TrainingDB

FROM DISK = 'C:\Backups\TrainingDB_Diff.bak'

WITH NORECOVERY;
```

-- 3. Restore TRANSACTION LOG backup (if you created one)

RESTORE LOG TrainingDB

FROM DISK = 'C:\Backups\TrainingDB_Log.trn'

WITH RECOVERY;

✅ Use WITH NORECOVERY until the final step.
✅ Use WITH RECOVERY only at the last step.

Step 3: Verify the Restored Data

USE TrainingDB;

SELECT * FROM Students;

Check if all the records (including the last one you added before the transaction log backup) are there.

Reflection Questions (Submit Answers):

1. What would happen if you skipped the **differential** backup step?
2. What's the difference between restoring a **full** vs. **copy-only** backup?
3. What happens if you use WITH RECOVERY in the middle of a restore chain?
4. Which backup types are **optional** and which are **mandatory** for full recovery?

Goal:

Show that you can **restore a SQL Server database using a backup chain**, and explain the logic of the steps you followed.