SQL Server Backups, Restores, and Security

Beginner Hands-On Lab Manual Using SSMS and T-SQL

Learn how to protect SQL Server data with backups, verify recovery with restores, and apply basic security using logins, users, roles, and permissions.

Audience	Beginners learning SQL Server administration basics
Estimated Time	2 to 3 hours
Tool	SQL Server Management Studio (SSMS)
Lab Database	AdminTrainingDB
Goal	Back up, restore, verify, and secure a practice database

Training Safety: Use a test SQL Server instance only. Do not practice restore or permission changes against production databases.

Lab Objectives

- Understand why backups are required and why restores must be tested.
- Identify full, differential, and transaction log backups at a beginner level.
- Understand simple and full recovery models.
- Create a practice database and perform full and differential backups.
- Restore a database under a different name for validation.
- Learn basic security concepts: authentication, authorization, logins, users, roles, and permissions.
- Create a least-privilege read-only user and a backup operator user.
- Review common restore security issues such as orphaned users.
- Complete challenge exercises and a final checklist.

Quick Concept Reference

Concept	Purpose	Beginner Meaning
Full backup	Baseline copy of the database	Required starting point for most restore plans
Differential backup	Changes since last full backup	Faster than another full in many cases
Log backup	Captures transaction log records	Supports point-in-time restore in full recovery model
Restore test	Validates backup usability	A backup is not trusted until restored successfully
Login	Instance-level authentication	Lets an account connect to SQL Server
User	Database-level identity	Maps access inside a database
Role	Permission group	Simplifies permission management

Module 1: Backup and Restore Fundamentals

Backups protect databases from user error, hardware failure, corruption, accidental object changes, and disaster scenarios. A complete strategy includes both backup jobs and restore testing.

- RPO: Recovery Point Objective. How much data loss is acceptable?
- RTO: Recovery Time Objective. How quickly must the service be restored?
- Backup chain: the ordered set of backups needed to restore to a target point.
- Restore validation: proving that backup files can actually recover the database.

Key idea: Backups are only half of the work. Restore testing proves whether the backup strategy works.

Module 2: Recovery Models

A recovery model controls transaction log behavior and the restore options available for a database.

Recovery Model	Log Backups?	Point-in-Time Restore?	Typical Beginner Use
Simple	No	No	Training, development, low-risk databases
Full	Yes	Yes	Production databases requiring minimal data loss
Bulk-logged	Yes	Limited during bulk operations	Special bulk load scenarios

```
-- View recovery model
SELECT name, recovery_model_desc
FROM sys.databases
WHERE name = 'AdminTrainingDB';

-- Change recovery model to SIMPLE for beginner backup practice
ALTER DATABASE AdminTrainingDB SET RECOVERY SIMPLE;
GO

-- Change recovery model to FULL when log backups are required
ALTER DATABASE AdminTrainingDB SET RECOVERY FULL;
GO
```

Important: If you use FULL recovery model, schedule transaction log backups. Without log backups, the transaction log can continue growing.

Module 3: Build the Training Database

Open SSMS, connect to a test SQL Server instance, select New Query, and run this setup script.

```
CREATE DATABASE AdminTrainingDB;
GO

USE AdminTrainingDB;
GO

CREATE TABLE dbo.SystemInventory
(
    AssetID INT IDENTITY(1,1) CONSTRAINT PK_SystemInventory PRIMARY KEY,
    ServerName VARCHAR(100) NOT NULL,
    Environment VARCHAR(30) NOT NULL,
    ApplicationName VARCHAR(100) NOT NULL,
    SupportTeam VARCHAR(100) NOT NULL,
    LastPatchDate DATE NULL
);
GO

INSERT INTO dbo.SystemInventory
    (ServerName, Environment, ApplicationName, SupportTeam, LastPatchDate)
VALUES
('APPWEB01', 'Production', 'IIS', 'ET Applications', '2026-07-01'),
('APPLNX01', 'Production', 'Apache', 'Linux Team', '2026-07-02'),
('SQLDB01', 'Production', 'SQL Server', 'Database Team', '2026-07-03'),
('APPTEST01', 'Test', 'Internal Test App', 'ET Applications', '2026-07-04');
GO
```

```
SELECT * FROM dbo.SystemInventory;
```

Module 4: Create Backups with SSMS and T-SQL

Lab 4.1: Prepare a Backup Folder

- Create a local training folder such as C:\SQLBackups.
- Make sure the SQL Server service account has permission to write to the backup folder.
- If SSMS cannot see the folder or backup fails with access denied, check service account file permissions.

Lab 4.2: Full Backup Using SSMS

- In Object Explorer, right-click AdminTrainingDB.
- Select Tasks > Back Up.
- Backup type: Full.
- Destination: C:\SQLBackups\AdminTrainingDB_Full.bak.
- Select OK and confirm the backup completes successfully.

Lab 4.3: Full Backup Using T-SQL

```
BACKUP DATABASE AdminTrainingDB
TO DISK = 'C:\SQLBackups\AdminTrainingDB_Full.bak'
WITH INIT,
     NAME = 'AdminTrainingDB Full Backup',
     STATS = 10;
GO
```

Lab 4.4: Differential Backup

```
USE AdminTrainingDB;
GO

INSERT INTO dbo.SystemInventory
    (ServerName, Environment, ApplicationName, SupportTeam, LastPatchDate)
VALUES
('DEVLNX01', 'Development', 'Batch Processor', 'Linux Team', '2026-07-05');
GO

BACKUP DATABASE AdminTrainingDB
TO DISK = 'C:\SQLBackups\AdminTrainingDB_Diff.bak'
WITH DIFFERENTIAL,
     INIT,
     NAME = 'AdminTrainingDB Differential Backup',
     STATS = 10;
GO
```

Module 5: Transaction Log Backup Practice

Transaction log backups require the FULL or BULK_LOGGED recovery model. This lab demonstrates the syntax in a training database.

```
ALTER DATABASE AdminTrainingDB SET RECOVERY FULL;
GO

-- Start a new full backup after switching to FULL recovery.
BACKUP DATABASE AdminTrainingDB
```

```
TO DISK = 'C:\SQLBackups\AdminTrainingDB_Full_ForLogs.bak'
WITH INIT, NAME = 'AdminTrainingDB Full Backup for Log Chain', STATS = 10;
GO

UPDATE dbo.SystemInventory
SET LastPatchDate = '2026-07-10'
WHERE ServerName = 'APPWEB01';
GO

BACKUP LOG AdminTrainingDB
TO DISK = 'C:\SQLBackups\AdminTrainingDB_Log.trn'
WITH INIT, NAME = 'AdminTrainingDB Transaction Log Backup', STATS = 10;
GO
```

Beginner note: In production, log backup frequency is based on business recovery requirements. Avoid switching models casually without understanding the backup chain.

Module 6: Restore and Verify Backups

A restore test proves the backup file is usable. Restore to a different database name so the original training database is not overwritten.

Lab 6.1: Inspect Backup Contents

```
RESTORE HEADERONLY
FROM DISK = 'C:\SQLBackups\AdminTrainingDB_Full.bak';
GO

RESTORE FILELISTONLY
FROM DISK = 'C:\SQLBackups\AdminTrainingDB_Full.bak';
GO
```

Lab 6.2: Restore Full Backup as a Test Database

```
RESTORE DATABASE AdminTrainingDB_RestoreTest
FROM DISK = 'C:\SQLBackups\AdminTrainingDB_Full.bak'
WITH
    MOVE 'AdminTrainingDB' TO 'C:\SQLData\AdminTrainingDB_RestoreTest.mdf',
    MOVE 'AdminTrainingDB_log' TO 'C:\SQLData\AdminTrainingDB_RestoreTest_log.ldf',
    REPLACE,
    STATS = 10;
GO

SELECT *
FROM AdminTrainingDB_RestoreTest.dbo.SystemInventory;
```

Lab 6.3: Restore Full plus Differential Backup

```
RESTORE DATABASE AdminTrainingDB_DiffRestore
FROM DISK = 'C:\SQLBackups\AdminTrainingDB_Full.bak'
WITH
    MOVE 'AdminTrainingDB' TO 'C:\SQLData\AdminTrainingDB_DiffRestore.mdf',
    MOVE 'AdminTrainingDB_log' TO 'C:\SQLData\AdminTrainingDB_DiffRestore_log.ldf',
    NORECOVERY,
    REPLACE,
    STATS = 10;
GO
```

```
RESTORE DATABASE AdminTrainingDB_DiffRestore
FROM DISK = 'C:\SQLBackups\AdminTrainingDB_Diff.bak'
WITH RECOVERY, STATS = 10;
GO

SELECT *
FROM AdminTrainingDB_DiffRestore.dbo.SystemInventory;
```

If logical file names are different in your environment, use RESTORE FILELISTONLY first and adjust the MOVE logical names.

Module 7: Security Fundamentals

SQL Server security separates authentication from authorization. Authentication determines whether an account can connect. Authorization determines what the account can do after connecting.

- Login: created at the SQL Server instance level.
- User: created inside a database and usually mapped to a login.
- Server role: grants broad instance-level powers. Use carefully.
- Database role: grants database-level permissions. Prefer roles over individual one-off grants.
- Least privilege: grant only the permissions required for a task.

Security rule: Do not use sysadmin for normal application access or routine reporting users.

Module 8: Create Logins, Users, Roles, and Permissions

This module creates a read-only training account and a role to show beginner-friendly permission design. Replace placeholder passwords with organization-approved secure values in a real environment.

```
USE master;
GO

-- Training example only. Replace with a secure password in real use.
CREATE LOGIN TrainingReadOnlyLogin
WITH PASSWORD = 'Replace_With_A_Strong_Training_Password_123!',
     CHECK_POLICY = ON,
     CHECK_EXPIRATION = ON;
GO

USE AdminTrainingDB;
GO

CREATE USER TrainingReadOnlyUser FOR LOGIN TrainingReadOnlyLogin;
GO

CREATE ROLE ReportingReaders;
GO

ALTER ROLE ReportingReaders ADD MEMBER TrainingReadOnlyUser;
GO

GRANT SELECT ON dbo.SystemInventory TO ReportingReaders;
GO
```

Lab 8.1: Verify Permissions

```
EXECUTE AS USER = 'TrainingReadOnlyUser';

SELECT * FROM dbo.SystemInventory;

-- This should fail because only SELECT was granted.
INSERT INTO dbo.SystemInventory
    (ServerName, Environment, ApplicationName, SupportTeam, LastPatchDate)
VALUES
('SECURITYTEST01', 'Test', 'Permission Test', 'Security Team', GETDATE());

REVERT;
GO
```

Module 9: Backup Operator Practice

The db_backupoperator database role can perform database backups for that database. Use carefully and only when the operational need exists.

```
USE master;
GO

CREATE LOGIN TrainingBackupLogin
WITH PASSWORD = 'Replace_With_A_Strong_Backup_Password_123!',
     CHECK_POLICY = ON,
     CHECK_EXPIRATION = ON;
GO

USE AdminTrainingDB;
GO

CREATE USER TrainingBackupUser FOR LOGIN TrainingBackupLogin;
GO

ALTER ROLE db_backupoperator ADD MEMBER TrainingBackupUser;
GO

-- Review role membership
SELECT
    dp1.name AS DatabaseRole,
    dp2.name AS DatabaseUser
FROM sys.database_role_members drm
INNER JOIN sys.database_principals dp1
    ON drm.role_principal_id = dp1.principal_id
INNER JOIN sys.database_principals dp2
    ON drm.member_principal_id = dp2.principal_id
WHERE dp2.name IN ('TrainingReadOnlyUser', 'TrainingBackupUser');
```

Even with SQL permissions, backup and restore actions also depend on file system permissions assigned to the SQL Server service account.

Module 10: Restore Security and Orphaned Users

When a database is restored to another SQL Server instance, database users may not match server logins if their security identifiers do not align. This is commonly called an orphaned user scenario.

```
-- Run inside the restored database to review users and login mappings.
USE AdminTrainingDB_RestoreTest;
GO

SELECT
    dp.name AS DatabaseUser,
    dp.type_desc AS UserType,
    sp.name AS ServerLogin
FROM sys.database_principals dp
LEFT JOIN sys.server_principals sp
    ON dp.sid = sp.sid
WHERE dp.type IN ('S', 'U', 'G')
  AND dp.principal_id > 4
ORDER BY dp.name;
GO

-- If a database user should map to an existing login, use:
-- ALTER USER [DatabaseUserName] WITH LOGIN = [LoginName];
```

Best practice: In migrations and refreshes, script required logins, users, roles, and permissions before restoring, then run a post-restore validation script.

Module 11: Backup, Restore, and Security Best Practices

- Document backup location, schedule, retention, owner, and restore procedure.
- Store a copy of backups away from the primary server or primary storage location.
- Test restores regularly and document restore results.
- Monitor backup job failures and storage capacity.
- Protect backup files because they contain database data.
- Use least privilege for logins, users, roles, and job accounts.
- Prefer database roles for permission management instead of many individual grants.
- Review sysadmin membership regularly.
- Script logins and permissions before migrations or environment refreshes.
- Validate application access after restores.

Challenge Exercises

#	Exercise	Done
1	Create a full backup of AdminTrainingDB using SSMS.	☐
2	Create a full backup using T-SQL with STATS = 10.	☐
3	Insert one new inventory row and create a differential backup.	☐
4	Restore the full backup as AdminTrainingDB_ChallengeRestore.	☐

#	Exercise	Done
5	Restore full plus differential backups as AdminTrainingDB_ChallengeDiffRestore.	☐
6	Use RESTORE HEADERONLY and RESTORE FILELISTONLY against your backup file.	☐
7	Create a read-only login, user, and role for the database.	☐
8	Verify the read-only user can SELECT but cannot INSERT.	☐
9	Create a backup operator user and review database role membership.	☐
10	Write a short paragraph describing your recommended backup schedule for a production application database.	☐

Mini Cheat Sheet

```
-- Full backup
BACKUP DATABASE DatabaseName
TO DISK = 'C:\SQLBackups\DatabaseName_Full.bak'
WITH INIT, STATS = 10;

-- Differential backup
BACKUP DATABASE DatabaseName
TO DISK = 'C:\SQLBackups\DatabaseName_Diff.bak'
WITH DIFFERENTIAL, INIT, STATS = 10;

-- Transaction log backup
BACKUP LOG DatabaseName
TO DISK = 'C:\SQLBackups\DatabaseName_Log.trn'
WITH INIT, STATS = 10;

-- Restore file list
RESTORE FILELISTONLY
FROM DISK = 'C:\SQLBackups\DatabaseName_Full.bak';

-- Create login, user, role, and grant SELECT
CREATE LOGIN LoginName WITH PASSWORD = 'StrongPasswordHere', CHECK_POLICY = ON;
USE DatabaseName;
CREATE USER UserName FOR LOGIN LoginName;
CREATE ROLE RoleName;
ALTER ROLE RoleName ADD MEMBER UserName;
GRANT SELECT ON dbo.TableName TO RoleName;
```

Final Lab Completion Checklist

Skill	Complete
Explained why backups and restore tests matter	☐
Identified full, differential, and log backups	☐
Reviewed simple and full recovery models	☐
Created AdminTrainingDB	☐
Created a full backup	☐
Created a differential backup	☐
Practiced transaction log backup syntax	☐
Used RESTORE HEADERONLY	☐
Used RESTORE FILELISTONLY	☐
Restored a database under a different name	☐
Created a login and database user	☐
Created a database role	☐
Granted SELECT using a role	☐
Validated read-only access	☐
Created backup operator user	☐
Reviewed orphaned user concept	☐
Completed challenge exercises	☐

References

Microsoft Learn: Back up and restore of SQL Server databases - https://learn.microsoft.com/en-us/sql/relational-databases/backup-restore/back-up-and-restore-of-sql-server-databases

Microsoft Learn: Restore database General page - https://learn.microsoft.com/en-us/sql/relational-databases/backup-restore/restore-database-general-page

Microsoft Learn: Recovery models - https://learn.microsoft.com/en-us/sql/relational-databases/backup-restore/recovery-models-sql-server

Microsoft Learn: Create a database user - https://learn.microsoft.com/en-us/sql/relational-databases/security/authentication-access/create-a-database-user

Microsoft Learn: Managing users, roles, and logins - https://learn.microsoft.com/en-us/sql/relational-databases/server-management-objects-smo/tasks/managing-users-roles-and-logins