TempDB vs RealDB: Performance Smackdown

-- =============================================  
-- 🧪 TempDB vs RealDB: Performance Smackdown  
-- =============================================  
-- Purpose: Run controlled performance comparisons between temp and real tables  
-- Author: Jimmy "Cage Match" Ruffles  
-- =============================================  
--  
-- 📝 INSTRUCTIONS FOR NON-SENIOR DBAs:  
-- This script will help you compare how SQL Server performs when using a temp table vs a regular table.  
-- Each section is labeled and explains what it does. Follow the steps in order.  
-- Make sure you replace [YourRealDB] with the name of your test database.  
-- You can run this script in SSMS (SQL Server Management Studio).  
-- Turn on 'Include Actual Execution Plan' and 'Include Client Statistics' for bonus insight.  
-- You don’t need to understand everything line-by-line. Just follow the comments and review the results.  
-- Ask a senior if you get weird errors or unexpected results.  
--  
-- 🔄 Summary of Steps:  
-- 1. Create and load a real table in your database.  
-- 2. Create and load a temp table in TempDB.  
-- 3. Add indexes to both tables.  
-- 4. Run a GROUP BY query on each to compare performance.  
-- 5. Run additional inserts into both tables.  
-- 6. Monitor TempDB if you want to see how much it's being used.  
-- =============================================  
  
-- Step 1: Setup Test Tables in RealDB  
USE [YourRealDB]; -- 👈 Replace with your database name  
GO  
  
IF OBJECT\_ID('dbo.RealTable') IS NOT NULL DROP TABLE dbo.RealTable;  
GO  
CREATE TABLE dbo.RealTable (  
 ID INT IDENTITY(1,1) PRIMARY KEY,  
 SomeData NVARCHAR(100),  
 SomeNumber INT  
);  
GO  
  
-- Step 2: Load RealTable with Data  
-- This inserts 100,000 rows using a system table to quickly generate data  
INSERT INTO dbo.RealTable (SomeData, SomeNumber)  
SELECT TOP 100000   
 CAST(NEWID() AS NVARCHAR(100)),   
 ABS(CHECKSUM(NEWID())) % 10000  
FROM sys.objects a CROSS JOIN sys.objects b;  
GO  
  
-- Step 3: Temp Table Creation and Load  
-- Temp tables live in TempDB and are dropped automatically when your session ends  
IF OBJECT\_ID('tempdb..#TempTable') IS NOT NULL DROP TABLE #TempTable;  
GO  
CREATE TABLE #TempTable (  
 ID INT IDENTITY(1,1) PRIMARY KEY,  
 SomeData NVARCHAR(100),  
 SomeNumber INT  
);  
  
-- Copy data from real table to temp table  
INSERT INTO #TempTable (SomeData, SomeNumber)  
SELECT SomeData, SomeNumber FROM dbo.RealTable;  
GO  
  
-- Step 4: Add Index to Both  
-- Index helps improve performance on GROUP BY and WHERE filters  
CREATE NONCLUSTERED INDEX IX\_RealTable\_SomeNumber ON dbo.RealTable(SomeNumber);  
CREATE NONCLUSTERED INDEX IX\_TempTable\_SomeNumber ON #TempTable(SomeNumber);  
GO  
  
-- Step 5: Run Performance Test - GROUP BY  
-- Real Table  
-- This groups the data by SomeNumber and counts rows in each group  
SET STATISTICS TIME ON;  
SELECT SomeNumber, COUNT(\*) FROM dbo.RealTable GROUP BY SomeNumber;  
SET STATISTICS TIME OFF;  
GO  
  
-- Temp Table  
-- Do the same for the temp table  
SET STATISTICS TIME ON;  
SELECT SomeNumber, COUNT(\*) FROM #TempTable GROUP BY SomeNumber;  
SET STATISTICS TIME OFF;  
GO  
  
-- Step 6: Run Insert Test  
-- We're simulating activity by inserting more rows  
-- Real Table Insert  
INSERT INTO dbo.RealTable (SomeData, SomeNumber)  
SELECT CAST(NEWID() AS NVARCHAR(100)), ABS(CHECKSUM(NEWID())) % 10000  
FROM sys.objects a CROSS JOIN sys.objects b;  
GO  
  
-- Temp Table Insert  
INSERT INTO #TempTable (SomeData, SomeNumber)  
SELECT CAST(NEWID() AS NVARCHAR(100)), ABS(CHECKSUM(NEWID())) % 10000  
FROM sys.objects a CROSS JOIN sys.objects b;  
GO  
  
-- Optional: Monitor TempDB Usage  
-- This shows how much space your temp objects are using in TempDB  
SELECT   
 SUM(user\_object\_reserved\_page\_count)\*8 AS usr\_obj\_kb,  
 SUM(internal\_object\_reserved\_page\_count)\*8 AS internal\_obj\_kb  
FROM sys.dm\_db\_file\_space\_usage;  
GO

# 💡 Lab Objective

In this lab, you'll compare performance between temporary tables and permanent tables in SQL Server.  
You'll observe CPU time, memory usage, and TempDB behavior under different workloads, using indexes and large data sets.

# 📦 Lab Setup Checklist

- ✅ You have access to a sandbox environment (NOT PRODUCTION).  
- ✅ You can modify TempDB file size if needed.  
- ✅ You understand how to monitor TempDB usage via DMV queries.

# 🧪 Pre-Lab Instruction

Before you begin:  
1. Ensure the TempDB file has enough space for repeated temp table activity.  
2. Optional: Cap TempDB at a specific size to simulate pressure conditions (e.g., MAXSIZE = 5120MB).  
 Example:

USE [master];  
GO  
ALTER DATABASE [tempdb] MODIFY FILE (  
 NAME = tempdev,  
 MAXSIZE = 5120MB  
);  
GO

# 📝 Lab Reflection Questions

After completing the steps, consider the following:  
- How did performance differ between the real and temp tables?  
- Did TempDB grow significantly?  
- What was the impact of adding indexes?  
- What happens when you drop and recreate the temp table in a loop?

# ✅ Lab Wrap-Up

You now have practical experience in benchmarking temp vs real table performance.  
This hands-on work is critical in understanding how SQL Server utilizes TempDB for internal and user objects.  
Be sure to clean up your test environment when done.