Create a test table with some random data

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:28 AM PST

Contents

Issue

This is a "How To" TSG with sample scripts for creating tables with test data.

Sample scripts

Single table with transaction batch control

The transaction batch control is important to allow for good performance. Without any transaction, each Insert represents a commit roundtrip to the transaction log, which is slow on Azure SQL Database. If you put everything into one single transaction, you might run out of transaction log space. Committing batches after e.g. every 100.000 rows is a good compromise for most scenarios.

```
CREATE TABLE [dbo].[TestTable] (
    [ID] [int] PRIMARY KEY CLUSTERED NOT NULL ,
    [c1] [varchar](100) NULL,
    [dt1] [datetime] DEFAULT GETDATE()
);

INSERT INTO [dbo].[TestTable] (ID, c1) VALUES (0, 'original insert');

SELECT * FROM [TestTable];


-- Insert a lot of rows in a loop
declare @i int = 1;

set nocount on

while @i <= 1000000
begin
    begin tran

    INSERT INTO [TestTable] (ID, c1) VALUES (@i, 'loop insert')

    -- commit every 100000 rows
    if @i % 100000 = 0
    begin
       commit tran
       begin tran
    end

    set @i += 1
end

set nocount off

-- Check number of rows and the overall execution time
SELECT count(*) FROM [TestTable]  -- 1000000
SELECT start_time = min(dt1), end_time = max(dt1), insert_duration = cast(max(dt1) - min(dt1) AS time) FROM [T


-- Cleanup
DROP TABLE [dbo].[TestTable];
```

Foreign Key tables with transaction batch control

See sample above regarding the importance of transaction batch control.

```
-- Primary Key table
CREATE TABLE [dbo].[MainTable] (
    [ID] [int] PRIMARY KEY CLUSTERED NOT NULL ,
    [c1] [varchar](100) NULL,
    [dt1] [datetime] DEFAULT GETDATE()
);
-- Foreign Key table
CREATE TABLE [dbo].[SubTable] (
    [ID] [int] PRIMARY KEY CLUSTERED NOT NULL,
    [MainTableID] [int] NOT NULL,
    [c1] [varchar](100) NULL,
    [dt1] [datetime] DEFAULT GETDATE()
    , CONSTRAINT FK_Main_Sub FOREIGN KEY (MainTableID) REFERENCES dbo.MainTable (ID) --NOT FOR REPLICATION
);
GO

INSERT INTO [dbo].[MainTable] (ID, c1) VALUES (0, 'original insert');
INSERT INTO [dbo].[SubTable] (ID, MainTableID, c1) VALUES (0, 0, 'original insert');

SELECT * FROM MainTable;
SELECT * FROM SubTable;

-- Insert a lot of rows in a loop
declare @outerloop int = 1;
declare @innerloop int = 1;
declare @i int = 1;

set nocount on

while @outerloop <= 100
begin
    set @innerloop = 1
    begin tran

    INSERT INTO [MainTable] (ID, c1) VALUES (@outerloop, 'loop insert')

    while @innerloop <= 1000
    begin
        INSERT INTO [SubTable] (ID, MainTableID, c1) VALUES (@i, @outerloop, 'loop insert')
        set @innerloop += 1
        set @i += 1
    end
    commit tran
    set @outerloop += 1
end

set nocount off


-- Check number of rows and the overall execution time
SELECT count(*) FROM [MainTable]  --    101
SELECT count(*) FROM [SubTable]   -- 100001
SELECT start_time = min(dt1), end_time = max(dt1), insert_duration = cast(max(dt1) - min(dt1) AS time) FROM [S


-- Cleanup
DROP TABLE [dbo].[SubTable];
DROP TABLE [dbo].[MainTable];
```

Dynamic column size, row size, and number of rows

```
-- Create sample table
CREATE TABLE [TestTable]
(
    ID int IDENTITY(1,1) PRIMARY KEY,
    data_1 varchar(8000) NULL,
    data_2 varchar(8000) NULL,
    data_3 varchar(max) NULL
);

-- Prepare the data to be inserted
DECLARE @i int
DECLARE @data1 varchar(8000)
DECLARE @data2 varchar(8000)
DECLARE @data3 varchar(8000)
SET @i = 0

-- Set column data length
WHILE (@i < 10)
BEGIN
    SET @data1 = CONCAT(@data1, 'A', CAST(@i AS varchar(3)))
    SET @data2 = CONCAT(@data2, 'B', CAST(@i AS varchar(3)))
    SET @data3 = CONCAT(@data3, 'C', CAST(@i AS varchar(3)))
    SET @i = @i + 1
END

-- Wrap the insert into a transaction to improve performance
begin tran
set nocount on

SET @i = 0
WHILE (@i < 10)
BEGIN
    INSERT INTO [TestTable](data_1, data_2, data_3) VALUES (@data1, @data2, @data3)
    SET @i = @i + 1
End

set nocount off
commit

-- Check results
select * FROM [TestTable]
/*
ID  data_1                data_2                data_3
--- --------------------- --------------------- --------------------
1   A0A1A2A3A4A5A6A7A8A9  B0B1B2B3B4B5B6B7B8B9  C0C1C2C3C4C5C6C7C8C9
2   A0A1A2A3A4A5A6A7A8A9  B0B1B2B3B4B5B6B7B8B9  C0C1C2C3C4C5C6C7C8C9
3   A0A1A2A3A4A5A6A7A8A9  B0B1B2B3B4B5B6B7B8B9  C0C1C2C3C4C5C6C7C8C9
4   A0A1A2A3A4A5A6A7A8A9  B0B1B2B3B4B5B6B7B8B9  C0C1C2C3C4C5C6C7C8C9
5   A0A1A2A3A4A5A6A7A8A9  B0B1B2B3B4B5B6B7B8B9  C0C1C2C3C4C5C6C7C8C9
(...)
*/

-- Cleanup
DROP TABLE [TestTable];
```

How good have you found this content?