Replicate partitioned tables Sample script

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:31 AM PST

Contents

Replicate partitioned tables - Sample script

This is a "How To" article demonstrating how partitioned tables can be used together with Transactional Replication.

Introduction

Partitioning makes large tables or indexes more manageable because partitioning enables you to manage and access subsets of data quickly and efficiently, and maintain the integrity of a data collection at the same time. The data of partitioned tables and indexes is divided into units and is partitioned horizontally, so that groups of rows are mapped into individual partitions. The table or index is treated as a single logical entity when queries or updates are performed on the data.

One of the key benefits of table partitioning is the ability to quickly and efficiently move subsets of data between partitions. Data is moved by using the SWITCH PARTITION command. This implies the following risks when using partitioning together with Transactional Replication:

- If data is moved into or out of a table that exists at the Publisher but does not exist at the Subscriber, the Publisher and Subscriber could become inconsistent with one another. This problem typically occurs when data is moved into or out of a staging table.
- If the Subscriber has a different definition for the partitioned table than the Publisher, the Distribution Agent will fail when it tries to apply changes at the Subscriber.

Despite these potential issues, partition switching can be enabled for Transactional Replication. Before you enable partition switching, make sure that all tables that are involved in partition switching exist at the Publisher and Subscriber, and make sure that the table and partition definitions are the same.

When partitions have the exact same partition scheme at the Publishers and Subscribers, you can turn on `@allow_partition_switch` along with `@replicate_partition_switch`, which will only replicate the partition switch statement to the Subscriber. You can also turn on `@allow_partition_switch` without replicating the DDL. This is

useful in the case where you want to roll old months out of the partition but keep the replicated partition in place for another year for backup purposes at the Subscriber.

Limitations are that changes to the partition scheme are not supported. Transactional replication doesn't support replicating the commands `ALTER PARTITION FUNCTION`, `ALTER PARTITION SCHEME`, or the `REBUILD WITH PARTITION` statement of `ALTER INDEX`. The changes associated with them will not be automatically replicated to the Subscriber. It is the responsibility of the user to make similar changes manually at the Subscriber.

Sample Script

Prerequisites before running the sample script

- Configure Distribution on your Managed Instance

Create databases, partition objects, partitioned tables

```
/** Start of Script **/

-- Create the publisher and subscriber databases
SET NOCOUNT ON
GO
USE [master]
GO
CREATE DATABASE [Repl_PUB]
GO
CREATE DATABASE [Repl_SUB]
GO

USE [Repl_PUB]
GO
CREATE USER TRANREPLADMIN FROM LOGIN TRANREPLADMIN
GO
EXEC sp_addrolemember N'db_owner', N'TRANREPLADMIN'
GO
USE [Repl_SUB]
GO
CREATE USER TRANREPLADMIN FROM LOGIN TRANREPLADMIN
GO
EXEC sp_addrolemember N'db_owner', N'TRANREPLADMIN'
GO
USE [master]
GO

-- Create the partition objects
USE [Repl_PUB]

CREATE PARTITION FUNCTION IntPartitionFunction(int)
    AS RANGE RIGHT FOR VALUES ( -1000, 1000, 2000)
GO
CREATE PARTITION SCHEME IntPartitionScheme
    AS  PARTITION IntPartitionFunction ALL TO ([PRIMARY])
GO
-- Partition scheme 'IntPartitionScheme' has been created successfully.
-- 'PRIMARY' is marked as the next used filegroup in partition scheme 'IntPartitionScheme'.

-- Create the sample tables and add a few rows
-- ReplTest is the main table
CREATE TABLE dbo.ReplTest
(
        [ID] INT NOT NULL,
        c1 VARCHAR(100) NOT NULL,
        dt1 DATETIME NOT NULL DEFAULT GETDATE()
        PRIMARY KEY CLUSTERED ([ID]) ON IntPartitionScheme([ID])
)
GO
-- ReplTest_backup is used for switching partitions
CREATE TABLE dbo.ReplTest_backup
(
        [ID] INT NOT NULL,
        c1 VARCHAR(100) NOT NULL,
        dt1 DATETIME NOT NULL DEFAULT GETDATE()
        PRIMARY KEY CLUSTERED ([ID]) ON IntPartitionScheme([ID])
)
GO
-- add sample data
declare @n int = -1001
begin tran
while @n < 3001
begin
     INSERT ReplTest ([ID], c1, dt1) VALUES (@n, 'original insert', GETDATE())
     set @n = @n + 1
end
commit
```

```
go
```

Create the publication and subscription

```
-- check if distribution has already been configured:
USE [master]
GO
exec sp_get_distributor
GO

-- Enable the replication database for publication
USE [master]
GO
exec sp_replicationdboption @dbname = N'Repl_PUB', @optname = N'publish', @value = N'true'
GO
-- configure the log reader agent
exec [Repl_PUB].sys.sp_addlogreader_agent @job_login = 'TRANREPLADMIN', @job_password = '$trongPa11word', @pub
     @publisher_login = 'TRANREPLADMIN', @publisher_password = '$trongPa11word'
GO

-- Add Publication
use [Repl_PUB]
exec sp_addpublication @publication = N'Publication_Tran', @description = N'Simple Transactional publication',
     @sync_method = N'native', @retention = 0,
     @allow_push = N'true', @allow_pull = N'true', @allow_anonymous = N'true', @enabled_for_internet = N'false
     @snapshot_in_defaultfolder = N'true', @compress_snapshot = N'false', @ftp_port = 21, @ftp_login = N'anony
     @allow_subscription_copy = N'false', @add_to_active_directory = N'false',
     @repl_freq = N'continuous', @status = N'active', @independent_agent = N'true',
     @immediate_sync = N'true',
     @allow_sync_tran = N'false', @autogen_sync_procs = N'false', @allow_queued_tran = N'false', @allow_dts =
     @replicate_ddl = 1, @allow_initialize_from_backup = N'false', @enabled_for_p2p = N'false', @enabled_for_h
    -- options for enabling partition handling:
    , @allow_partition_switch = 'true', @replicate_partition_switch = 'true'
GO
-- Configure the Snapshot Agent
-- (sets an inactive schedule so that you can start it manually for your tests)
exec sp_addpublication_snapshot @publication = N'Publication_Tran',
     @frequency_type = 1, @frequency_interval = 0, @frequency_relative_interval = 0, @frequency_recurrence_fac
     @active_start_time_of_day = 0, @active_end_time_of_day = 235959,
     @active_start_date = 0, @active_end_date = 20051231,
     @job_login = 'TRANREPLADMIN', @job_password = '$trongPa11word',
     @publisher_security_mode = 0, @publisher_login = 'TRANREPLADMIN', @publisher_password = '$trongPa11word'
GO
-- Grant permissions to publication
exec sp_grant_publication_access @publication = N'Publication_Tran', @login = N'sa'
GO
exec sp_grant_publication_access @publication = N'Publication_Tran', @login = N'TRANREPLADMIN'
GO

-- Adding the transactional articles
use [Repl_PUB]
exec sp_addarticle @publication = N'Publication_Tran',
        @article = N'ReplTest',
        @source_owner = N'dbo', @source_object = N'ReplTest',
        @destination_owner = N'dbo', @destination_table = N'ReplTest',
        @type = N'logbased', @description = N'', @creation_script = N'',
        @pre_creation_cmd = N'drop',
        @schema_option = 0x00000000081F509F, @identityrangemanagementoption = N'none',
--  @schema_option = 0x000000000803529F, @identityrangemanagementoption = N'none',
        @status = 24, @vertical_partition = N'false',
        @ins_cmd = N'CALL [sp_MSins_dboReplTest]', @del_cmd = N'CALL [sp_MSdel_dboReplTest]', @upd_cmd = N'SCA
GO
exec sp_addarticle @publication = N'Publication_Tran',
        @article = N'ReplTest_backup',
        @source_owner = N'dbo', @source_object = N'ReplTest_backup',
        @destination_owner = N'dbo', @destination_table = N'ReplTest_backup',
        @type = N'logbased', @description = N'', @creation_script = N'',
        @pre_creation_cmd = N'drop',
        @schema_option = 0x00000000081F509F, @identityrangemanagementoption = N'none',
--  @schema_option = 0x000000000803529F, @identityrangemanagementoption = N'none',
        @status = 24, @vertical_partition = N'false',
        @ins_cmd = N'CALL [sp_MSins_dboReplTest_backup]', @del_cmd = N'CALL [sp_MSdel_dboReplTest_backup]', @u
```

```
GO

-- Add the subscription
-- This is a Push subscription into a Managed Instance (using the same instance as the publisher)
use [Repl_PUB]
exec sp_addsubscription @publication = N'Publication_Tran',
     @subscriber = @@servername, @destination_db = N'Repl_SUB',
     @subscription_type = N'Push', @sync_type = N'automatic', @article = N'all',
     @update_mode = N'read only', @subscriber_type = 0
GO
-- Configure the Distribution Agent
exec sp_addpushsubscription_agent @publication = N'Publication_Tran',
     @subscriber = @@servername, @subscriber_db = N'Repl_SUB',
     @job_login = 'TRANREPLADMIN', @job_password = '$trongPa11word', @subscriber_security_mode = 0, @subscribe
     @frequency_type = 64, @frequency_interval = 0, @frequency_relative_interval = 0, @frequency_recurrence_fa
--     @frequency_type = 4, @frequency_interval = 1, @frequency_relative_interval = 0, @frequency_recurrence_f
GO

-- Check Replication Monitor here for success agent execution
```

Test the SWITCH PARTITION functionality

```
-- Check the initial data distribution
select count(*) from [Repl_PUB]..ReplTest -- 4002
select count(*) from [Repl_SUB]..ReplTest -- 4002
select count(*) from [Repl_PUB]..ReplTest_backup -- 0
select count(*) from [Repl_SUB]..ReplTest_backup -- 0

-- Move all partitions from ReplTest to ReplTest_backup

use [Repl_PUB];
ALTER TABLE ReplTest SWITCH PARTITION 1 to ReplTest_backup PARTITION 1;

/* DON'T DO THIS:
ALTER TABLE [Repl_PUB]..ReplTest SWITCH PARTITION  1 to [Repl_PUB]..ReplTest_backup PARTITION 1

Will fail with:
-- ALTER TABLE SWITCH statement failed.
-- The source table 'ReplTest' is in database 'Repl_SUB' while
-- the target table 'Repl_PUB..ReplTest_backup' is in database 'Repl_PUB'.
-- (Source: MSSQLServer, Error number: 4948)
*/

-- Check data distribution
select count(*) from [Repl_PUB]..ReplTest -- 4001
select count(*) from [Repl_SUB]..ReplTest -- 4001
select count(*) from [Repl_PUB]..ReplTest_backup -- 1
select count(*) from [Repl_SUB]..ReplTest_backup -- 1

use [Repl_PUB];
ALTER TABLE ReplTest SWITCH PARTITION  2 to ReplTest_backup PARTITION 2;

-- Check data distribution
select count(*) from [Repl_PUB]..ReplTest -- 2001
select count(*) from [Repl_SUB]..ReplTest -- 2001
select count(*) from [Repl_PUB]..ReplTest_backup -- 2001
select count(*) from [Repl_SUB]..ReplTest_backup -- 2001

use [Repl_PUB];
ALTER TABLE ReplTest SWITCH PARTITION  3 to ReplTest_backup PARTITION 3;

-- Check data distribution
select count(*) from [Repl_PUB]..ReplTest -- 1001
select count(*) from [Repl_SUB]..ReplTest -- 1001
select count(*) from [Repl_PUB]..ReplTest_backup -- 3001
select count(*) from [Repl_SUB]..ReplTest_backup -- 3001


use [Repl_PUB];
ALTER TABLE ReplTest SWITCH PARTITION  4 to ReplTest_backup PARTITION 4;

-- Check data distribution
select count(*) from [Repl_PUB]..ReplTest -- 0
select count(*) from [Repl_SUB]..ReplTest -- 0
select count(*) from [Repl_PUB]..ReplTest_backup -- 4002
select count(*) from [Repl_SUB]..ReplTest_backup -- 4002
-- no data in ReplTest
-- all data has moved to ReplTest_backup

-- Check output of Distribution Agent:
select * from distribution..MSdistribution_history order by time desc
-- A DDL change has been replicated.
-- A DDL change has been replicated.
-- A DDL change has been replicated.
-- A DDL change has been replicated.
-- 1 transaction(s) with 1 command(s) were delivered.
```

Deomonstrate ALTER PARTITION FUNCTION

```
/*** NOTE THAT THE FOLLOWING IS NOT SUPPORTED BY REPLICATION: ********************/
-- ALTER PARTITION FUNCTION IntPartitionFunction() MERGE RANGE (-1000)
-- -> SQL Server does not provide replication support for modifying a partition function.
-- -> Changes to a partition function in the publication database must be manually applied in the subscription

-- Check current partition ranges:
select * from [Repl_PUB].sys.partition_range_values
select * from [Repl_SUB].sys.partition_range_values
/*
function_id  boundary_id  parameter_id  value
65536        1            1             -1000
65536        2            1             1000
65536        3            1             2000
*/

select * from [Repl_PUB].sys.partition_functions
select * from [Repl_SUB].sys.partition_functions
/*
name                  function_id  type  type_desc  fanout  boundary_value_on_right  is_system
IntPartitionFunction  65536        R     RANGE      4       1                        0
*/


-- Quiesce the topology before running the following
-- no other workloads on the Publisher database and all pending changes were replicated to the Subscriber
use [Repl_PUB]
ALTER PARTITION FUNCTION IntPartitionFunction() MERGE RANGE (-1000)
ALTER PARTITION SCHEME IntPartitionScheme NEXT USED [Primary];
ALTER PARTITION FUNCTION IntPartitionFunction() SPLIT RANGE (3000)

use [Repl_SUB]
ALTER PARTITION FUNCTION IntPartitionFunction() MERGE RANGE (-1000)
ALTER PARTITION SCHEME IntPartitionScheme NEXT USED [Primary];
ALTER PARTITION FUNCTION IntPartitionFunction() SPLIT RANGE (3000)

use [Repl_PUB]

select * from [Repl_PUB].sys.partition_range_values
select * from [Repl_SUB].sys.partition_range_values
/*
function_id  boundary_id  parameter_id  value
65536        1            1             1000
65536        2            1             2000
65536        3            1             3000
*/

-- Insert more data into the Publisher table
declare @n int = 3001
begin tran
while @n < 5001
begin
     INSERT [Repl_PUB]..ReplTest ([ID], c1, dt1) VALUES (@n, 'later insert', getdate())
     set @n = @n + 1
end
commit
go

-- Check data distribution
select count(*) from [Repl_PUB]..ReplTest -- 2000
select count(*) from [Repl_SUB]..ReplTest -- 2000
select count(*) from [Repl_PUB]..ReplTest_backup -- 4002
select count(*) from [Repl_SUB]..ReplTest_backup -- 4002


/** End of Script **/
```

Public Doc Reference

- Replicate Partitioned Tables and Indexes
- Partitioned tables and indexes
- Switching partitions between tables
- Using SWITCH to move a partition to a history table

How good have you found this content?