

CDC Change Table cleanup custom procedure


Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:30 AM PST

Contents

- [Issue](#)
- [Investigation / Analysis](#)
- [Mitigation](#)
- [Internal reference](#)

Issue

This article addresses the situation where a _CT table is so big that its cleanup never completes within the given time period. In the specific customer scenario, when the regular CDC cleanup job started, it first cleaned up all other _CT tables before coming around to the larger tables. The customer had to stop the regular CDC cleanup job during daytime because it affected the production workload (blocking of the capture job, thus filling the transaction log and causing MI to run out of storage).

The custom cleanup procedure in the "Mitigation" section below focuses on a specific table, thus allowing for targeted cleanup of larger tables. The actual cleanup is done through the normal [sys.sp_cdc_cleanup_change_table \(Transact-SQL\)](#)  system procedure.

Investigation / Analysis

For a discussion regarding the technical background and other mitigation options, see [CDC Change Tables cleanup slow or not working](#).

The stored procedure shown in the "Mitigation" section below relates to "Mitigation 2" in that article. Although it is using the same call to `sys.sp_cdc_cleanup_change_table`, the custom procedure below has much better logging options, error handling, some and sanity checks. If this is not a one-time incident, then the recommendation is to use the custom cleanup procedure.

Mitigation

This procedure has been provided by PG on a support case.

```

CREATE OR ALTER PROCEDURE [dbo].[custom_cdc_ct_table_cleanup] (@capture_instance_name varchar(200), @LogToTable
as
begin
-- step 0, declarations
declare @low_water_mark_time datetime
, @low_water_mark binary(10)
, @last_commit_time datetime
, @retention bigint
, @threshold bigint
, @startTime datetime = GETDATE()
, @retcode char(10)
, @cleanup_failed_bit bit = 0

-- create table to store the history if logging is enabled
IF object_id('custom_cdc_cleanup_history') is null and @LogToTable=1
BEGIN
    CREATE TABLE [dbo].[custom_cdc_cleanup_history] (
        start_time datetime not null
        , end_time datetime
        , capture_instance_name varchar(200) not null
        , last_commit_time datetime
        , low_water_mark_time datetime
        , retention bigint
        , threshold bigint
        , max_retries smallint
        , retries smallint default 0
        , result char(10));
END

-- step 1, retention and threshold
-- taking retention and threshold from the CDC setup
-- (make sure threshold is below 5000 to avoid lock escalation)
select @retention = retention, @threshold = threshold from msdb.dbo.cdc_jobs_view where job_type = N'cleanup'

-- step 2, last commit time
set @last_commit_time = (select max(tran_end_time) from cdc.lsn_time_mapping)

-- step 3, retention time frame
-- subtract retention from the last commit time
set @low_water_mark_time = dateadd(minute, -@retention, @last_commit_time)

-- step 4, retention time frame adjustment
-- recompute / adjust the time to the closest LSN time
select @low_water_mark_time = sys.fn_cdc_map_lsn_to_time (sys.fn_cdc_map_time_to_lsn('largest less than or equ

-- step 5, selection of the exact watermark LSN
-- find the min LSN for the that particular time instance (as there could be more LSNs for it)
select @low_water_mark = min(start_lsn) from [cdc].[lsn_time_mapping] where tran_end_time = @low_water_mark_ti

if @LogToTable=1 insert into [dbo].[custom_cdc_cleanup_history] (start_time, capture_instance_name, last_commit_time,
select @startTime, @capture_instance_name, @last_commit_time, @low_water_mark_time, @retention, @threshold, @M

declare @attempt smallint = 0

while (@attempt <= @MaxRetries)
begin
-- step 6, the cleanup
-- cleans from the tracking table and the cdc.lsn_time_mapping table
set @cleanup_failed_bit = 0

EXEC @retcode = sys.sp_cdc_cleanup_change_table
    @capture_instance = @capture_instance_name, -- change the capture instance (table) name in this line
    @low_water_mark = @low_water_mark,
    @threshold = @threshold,
    @fCleanupFailed = @cleanup_failed_bit OUTPUT;

-- Write to the history table.
if @LogToTable=1 update [dbo].[custom_cdc_cleanup_history] set retries=@attempt

```


```
where start_time=@startTime and capture_instance_name=@capture_instance_name;

if @cleanup_failed_bit > 0
begin
    -- The following delay gives any other transactions
    -- that might have caused the cleanup to fail
    -- a chance to complete and release its locks
    waitfor delay '00:00:05'
    set @attempt += 1
end
else
begin
    if @LogToTable=1 update [dbo].[custom_cdc_cleanup_history] set end_time=GETDATE(), result='Success'
    where start_time=@startTime and capture_instance_name=@capture_instance_name;
    return
end
end

if @LogToTable=1 update [dbo].[custom_cdc_cleanup_history] set end_time=GETDATE(), result='Failure'
where start_time=@startTime and capture_instance_name=@capture_instance_name;

end
GO
```

Internal reference

- [CDC Change Tables cleanup slow or not working](#)
- Support case 2207260050000930
- [lcM 323219139](#) 

How good have you found this content?

