


# CDC Add a new column on the base table

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

## Contents

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

## Issue

Customer wants to add a new column on a source table. Since CDC doesn't reflect any DDL changes, the customer followed a workaround found on the [internet](#) . This creative but totally unsupported approach basically creates a second capture instance for the source table with `sys.sp_cdc_enable_table`, manually copies all rows from the old `_CT` table into the new `_CT` table and then manually updates the `[cdc].[change_tables]` to the old `start_lsn`.

Now CDC is failing with a set of errors reported to `sys.dm_cdc_errors`:

Error 22859: Log Scan process failed in processing log records. Refer to previous errors in the current session to identify the cause and correct any associated problems.

Error 18805: The Log-Scan Process failed to construct a replicated command from log sequence number (LSN) **{00015b5b:00003230:0139}**. Back up the publication database and contact Customer Support Services.


Error 2627: Violation of PRIMARY KEY constraint 'lsn\_time\_mapping\_clustered\_idx'. Cannot insert duplicate key in object 'cdc.lsn\_time\_mapping'. The duplicate key value is **(0x00015ae100005f100001)**.

The comment section of the same article also reports a different CDC failure after following these unsupported steps:

It looked like everything worked nicely, however CDC would then refuse to put new entries in the updated capture instance. I could not see any problem anywhere – even disabling CDC on the table and re-enabling CDC didn't fix it.

I ended up contacting MS support who were also unable to repair whatever damage I'd caused. The final fix suggested was to disable CDC on the entire database and re-enable CDC on every table, which of course lost all my updates.

## Investigation / Analysis

The error can be seen from customer side, querying [sys.dm\\_cdc\\_errors](#) . They are also logged in Kusto's MonCDCTraces.

Note also that the workaround followed by the customer is not supported.

## Mitigation

On this case, a mitigation was attempted by removing the duplicate key entries from the `cdc.lsn_time_mapping` system table. Delete the existing `start_lsn` that is conflicting:

```
select * from cdc.lsn_time_mapping where start_lsn = 0x00015ae100005f100001
```


start_lsn	tran_begin_time	tran_end_time	tran_id	tran_begin_lsn
0x00015AE100005F100001	2022-06-05 15:40:30.820	2022-06-05 15:40:30.820	0x00	0x00000000000000000000

```
DELETE from cdc.lsn_time_mapping where [start_lsn] = 0x00015ae100005f100001
```

After this, restart CDC. You might have to repeat this step for different LSNs if it continues to fail.

In this specific case, however, the issue persisted; the same error for the same primary key was occurring even after the row was deleted from the `cdc.lsn_time_mapping` table. We suggested to the customer to disable and re-enable CDC, with the loss of all change data.

## More Information

Again, note that the procedure taken by the customer is not supported. Check the supported method at [Handling changes to source table](#) .

Although enabling change data capture on a source table does not prevent such DDL changes from occurring, change data capture helps to mitigate the effect on consumers by allowing the delivered result sets that are returned through the API to remain unchanged even as the column structure of the underlying source table changes. This fixed column structure is also reflected in the underlying change table that the defined query functions access.

...the capture process responsible for populating the change table will ignore any new columns that are not identified for capture when the source table was enabled for change data capture. If a tracked column is dropped, null values will be supplied for the column in the subsequent change entries. However, if an existing column undergoes a change in its data type, the change is propagated to the change table to ensure that the capture mechanism does not introduce data loss to tracked columns.

Typically, the current capture instance will continue to retain its shape when DDL changes are applied to its associated source table. However, it is possible to create a second capture instance for the table that reflects the new column structure. This allows the capture process to make changes to the same source table into two distinct change tables having two different column structures. Thus, while one change table can continue to feed current operational programs, the second one can drive a development environment that is trying to incorporate the new column data. Allowing the capture mechanism to populate both change tables in tandem means that a transition from one to the other can be accomplished without loss of change data. This can happen anytime the two change data capture timelines overlap. When the transition is affected, the obsolete capture instance can be removed.

## Internal reference

[lcM 314579227](#) 

**How good have you found this content?**

