

SQLskills Immersion Event

Capital IQ Onsite Engagement

Module 8 (Extra): Change Data Capture

Jonathan Kehayias

Jonathan@SQLskills.com



Overview

- **Change identification techniques**
- **Configuring Change Data Capture**
- **Identifying and consuming changes in change tables**
- **Handling schema changes**
- **Performance and feature considerations**
- **SQL Server 2012 SSIS components**

Techniques for Identifying Changes

- **DML table triggers**
 - Can track before and after row state and deleted rows
 - Can be customized to include the user, modification time, or input buffer
 - Can introduce significant performance overhead on transactional systems
- **Modified datetime or timestamp column**
 - Can introduce performance overhead to pull changes
 - Does not track deleted rows
 - Requires schema modifications to source tables and code to set value
- **Data comparisons**
 - Comparing source and destination data requires scanning all rows to determine changes and introduces performance significant overhead
- **Replication and Subscriber triggers**
 - Offloads change identification to subscriber database
 - Requires customizations and manual management of schema changes

The Change Data Capture Solution

- **Change Data Capture provides information about DML changes to a table in near real-time using the same Log Reader Agent as transactional replication**
 - Eliminates expensive techniques that require schema modifications
 - DML triggers
 - Timestamp columns
 - Data comparisons and complex JOIN queries
- **May be used to answer a variety of critical questions:**
 - What are all of the changes that happened to a table since the last ETL?
 - Which columns changed?
 - What type of changes occurred? INSERT/UPDATE/DELETE?
 - What was the before image of a row that was modified?
- **Supports net change identification, with a performance trade-off for maintaining an additional nonclustered index on change tables**

Change Data Capture Requirements

- **Enterprise Edition only**
- **Enabling CDC for a database requires sysadmin privileges**
 - Requires executing `sp_cdc_enable_db`
- **Enabling CDC for a table requires db_owner privileges in the database**
 - Requires executing `sp_cdc_enable_table` for each table that will track changes within the database
- **Querying the results from the CDC tables requires membership within the database role specified in the `sp_cdc_enable_table` procedure call if specified**

Demo

Intro demo: Enabling Change Data Capture

sp_cdc_enable_table

- **@source_schema** – the name of the source table schema (required)
- **@source_name** – the name of the source table (required)
- **@role_name** – the name of the database security role used for granting access to the change data (required but can explicitly be set to NULL)
- **@capture_instance** – the name of the capture instance (optional)
- **@supports_net_changes** – indicates whether querying for net changes is supported by the capture instance (optional defaults 1)
- **@index_name** – the name of a unique index to uniquely identify rows in the source table (optional defaults to primary key)
- **@captured_column_list** – the list of source table columns to include in the change table (optional)
- **@filegroup_name** – the filegroup to be used for the change table (optional)
- **@allow_partition_switch** – indicates whether the SWITCH PARTITION command of ALTER TABLE can be executed against a table that is enabled for CDC (optional)

Demo

Enabling Tables for Change Data Capture

Change Data Capture Agent Jobs

- **Two jobs are created in SQL Server Agent when a database is enabled for CDC and the first table is enabled for tracking**
 - Capture job – reads transaction log records and populates change tables
 - The capture job is only created if no transactional replication publications exist for the CDC-enabled database
 - The capture job is created when CDC and transactional replication are enabled for a database, and the logreader job is removed when the database is no longer a Publisher
 - Cleanup job – removes entries from change tables older than the retention period specified for instance
 - The cleanup job is always created when the first table is enabled
- **The CDC Agent jobs are removed when CDC is disabled for a database**
 - The capture job can also be removed when the first publication is added to a database, and both CDC and replication are enabled

CDC Agent Job Options

- The CDC Agent jobs can be modified using the stored procedure `sys.sp_cdc_change_job` and specifying the appropriate job type to modify
- **'capture' job type parameters**
 - `@maxtrans` – maximum number of transactions to process in a single scan
 - `@maxscans` – maximum number of scan cycles to process rows from the log
 - `@continuous` – indicates whether the job runs continuously or exits after the maximum scan cycles configured for `@maxscans`
 - `@pollinginterval` – the number of seconds between log cycle scans
- **'cleanup' job type parameters**
 - `@retention` – number of minutes to retain data in change tables (default 3 days, maximum 100 years)
 - `@threshold` – maximum number of entries that can be deleted using a single statement on cleanup (default 5000)
- **Configured values can be determined using `sys.sp_cdc_help_jobs`**

Demo

SQL Agent Jobs for Change Data Capture

Change Table Columns

- **Change tables exist within the CDC schema and are named <capture_instance>_CT**
- **The first five columns are metadata columns:**
 - __\$start_lsn – the starting LSN of the transaction
 - __\$end_lsn – the ending LSN of the transaction
 - __\$seqval – the sequence or order of the row changes within a transaction
 - __\$operation – the type of operation reflected by the change row
 - 1 = Delete
 - 2 = Insert
 - 3 = Value before update
 - 4 = Value after update
 - __\$update_mask – bitmask of columns changed by operation within row
- **Remaining columns match the source table column definition when the capture instance was created**

Demo

Change Tables in the CDC Schema

Change Row Table-Valued Functions

- **cdc.fn_cdc_get_all_changes_<capture_instance>**
 - Returns one row for each modification applied to the source table within the specified LSN range
 - Multiple modifications of a source row within the LSN range will be represented individually in the result set
- **cdc.fn_cdc_get_net_changes_<capture_instance>**
 - Returns a single net change row for each source row modified within the specified LSN range

Determining Change Rows to Process

■ By LSN:

- `sys.fn_cdc_map_time_to_lsn ( '<relational_operator>', tracking_time )`
 - Returns the LSN value from the `start_lsn` column of the `cdc.lsn_time_mapping` table for the `tracking_time` specified
 - The `relational_operator` specifies the comparison to be applied against the `tran_end_time` of the `cdc.lsn_time_mapping` table when determining the LSN value to return
 - largest less than
 - largest less than or equal
 - smallest greater than
 - smallest greater than or equal
- `sys.fn_cdc_get_min_lsn ( 'capture_instance_name' )`
 - Returns the `start_lsn` value for the capture instance from `cdc.change_tables`
 - Sets the lower endpoint for change data for a given capture instance
- `sys.fn_cdc_get_max_lsn ( )`
 - Returns the maximum `start_lsn` column value from `cdc.lsn_time_mapping` setting the high endpoint for all capture instances

Determining Change Rows to Process (2)

■ By LSN:

- Custom tracking table updated by application code to track the capture instance name and last processed LSN
- `sys.fn_cdc_decrement_lsn ( lsn_value )`
 - Returns the previous LSN in the sequence based upon the specified LSN
 - Often used to decrement `sys.fn_cdc_get_max_lsn()` value to set high endpoint without overlapping LSNs across different data loads
- `sys.fn_cdc_increment_lsn ( lsn_value )`
 - Returns the next LSN in the sequence based upon the specified LSN
 - Often used to increment the last saved LSN from a custom tracking table to set a new lower endpoint without overlapping LSNs across different data loads

■ By time:

- `sys.fn_cdc_map_lsn_to_time ( lsn_value )`
 - Returns the `tran_end_time` column from `cdc.lsn_time_mapping` for the specified LSN, allowing LSN ranges to be mapped to specific time ranges

Demo

Using Change Rows Table-Valued Functions

Determining Whether a Column Changed

- **sys.fn_cdc_get_column_ordinal ('capture_instance', 'column_name')**
 - Returns the ordinal position of a column name within the specified capture instances update mask
- **sys.fn_cdc_is_bit_set (position, update_mask)**
 - Checks the specified ordinal position of the update mask to determine if the change bit is set
- **sys.fn_cdc_has_column_changed ('capture_instance', 'column_name' , update_mask)**
 - Identifies whether the specified column has been updated in the associated change row
 - Ideally only used for post processing – use sys.fn_cdc_get_column_ordinal once to set the position, and sys.fn_cdc_is_bit_set to parse the update_mask in queries against change tables for better performance

Demo

Determining Column Changes in Changed Rows

Handling Schema Changes with CDC

- **DDL changes to source tables are tracked by CDC, but not applied automatically to change tables in the CDC schema**
 - Automatically applying changes could break ETL processes or destinations not modified separately to handle the metadata change
 - Changes continue to be captured with new source columns dropped
- **CDC supports up to two capture instances per source table**
 - Creating a new capture instance after changing the table DDL provides support for existing consumers of change table data, and updated consumers of the new DDL
 - Removing the oldest capture instance is recommended once applications have been updated to use the new capture instance and columns
- **The complete DDL change history for a capture instance can be found using the `sys.sp_cdc_get_ddl_history` procedure**

Demo

Handling Schema Changes with CDC

Change Table Clean-Up

- **The size of the change tables for CDC would grow out of control without a programmatic/systematic cleanup process in place**
 - The default cleanup job removes entries older than three days from the change tables nightly at 2AM
 - The default retention period, in minutes, can be modified using `sys.sp_cdc_change_job` for the cleanup job in the database
- **Manual cleanup of change tables can be handled by ETL processes for more direct control over the cleanup process by executing the `sys.sp_cdc_cleanup_change_table` procedure for the capture instance after processing the change rows**

Demo

Controlling Change Table Clean-Up

Performance Considerations

- **Create change tables in a separate filegroup dedicated to CDC objects only – allows for isolation of I/O workload**
- **Limit capture columns to the minimum required set of columns to meet ETL requirements from change tables**
 - For complex ETL processes requiring a query against the source system and JOINS to multiple tables, only track primary keys on required tables
- **Only use the net changes functionality where absolutely required or on tables with relatively small volumes of data modifications**
 - Net changes require maintaining an additional nonclustered index on the change table, introducing additional overhead

Restore Considerations

- Restoring a CDC enabled database will remove CDC from the database when WITH RECOVERY is specified by the RESTORE operation by default
- The KEEP_CDC option for RESTORE will allow the database to be recovered while maintaining the current CDC configuration and data from the current recovery point in time for the database
 - The capture and cleanup jobs must be created manually after restoring a CDC enabled database using WITH KEEP_CDC using sys.sp_cdc_add_job
- ETL processes and other downstream consumers of CDC data should be designed to log the last LSN processed from capture tables
 - Identify cases where source database is restored to an earlier point in time
 - Identify where to restart processing from source database that is restored to a later point in time within the LSN interval window of CDC retention

CDC Feature Compatibility

- **Database Mirroring and Availability Groups**
 - The capture and cleanup job will not exist on the mirror server and must be created using `sp_cdc_add_job` after a failover occurs
- **Replication**
 - The capture job is deleted when replication is configured for the source database and the logreader Agent captures changes for replication and CDC
 - All changes are first written to the distribution database by the logreader Agent, then to the CDC change tables
 - If there is latency writing to the distribution database it will affect CDC change table population as well
 - Stored Procedure execution replication is disallowed when CDC is enabled for a database
- **CDC is not supported for Partially Contained Databases**

SQL Server 2012 SSIS Components

- Prior to SQL Server 2012, custom ETL processes had to be designed to read CDC change tables and load destination databases with results
- SQL Server 2012 includes the Attunity CDC SSIS tasks which simplify interacting with change tables
 - CDC Control Task – controls the life cycle of CDC packages, by handling the synchronization of an initial load, the LSN ranges processed by incremental load executions, and any error or recovery scenarios
 - CDC Source – reads a range of change rows from the CDC change tables and delivers the rows to other SSIS components in the data flow tasks
 - CDC Splitter – splits a single flow of change rows from the CDC Source component into separate data flows for INSERT, UPDATE and DELETE operations based on the __\$operation column values read by the CDC Source

Demo

SQL Server 2012 SSIS Components

Review

- Change identification techniques
- Configuring Change Data Capture
- Identifying and consuming changes in change tables
- Handling schema changes
- Performance and feature considerations
- SQL Server 2012 SSIS components

Questions?

