http://www.sqlservercentral.com/articles/Change+Data+Capture+(CDC)/64289/
Printed 2008/10/24 09:14AM

Implementing Change Data Capture in Microsoft® SQL Server 2008

By Suresh Babu Yaram, 2008/09/07

Abstract

Our ability to track changes to our data effectively is critical in refreshing the data from the source systems in data warehousing applications. In earlier days, the change tracking mechanism was often implemented by simply using the 'Last Updated DATETIME' column in the source system tables to determine the rows that have been changed since the previous refresh. While this is simple and pretty effective in determining the newly Inserted/Updated data, it is of no use in determining the rows that were physically deleted from the table. Besides this, we can't determine what data was changed when; we can only access the current state of a row that has changed. In order to maintain the before and the after image of data, one has to implement this mechanism with the help of triggers, requiring a lot of development effort. Change Data Capture (CDC), which is new in SQL Server 2008, provides a configurable solution that addresses these requirements. We could consider CDC to reduce our ETL time in data integration projects requiring the regular extraction and loading of a huge amount of data from one system to another system or systems.

Introduction

Change Data Capture (CDC) is a new feature available in the Microsoft SQL Server 2008 Enterprise & Developer versions. This feature facilitates the ability to record all data modification activities such as INSERT, UPDATE, and DELETE as applied to SQL Server tables in an easily consumable format. The column information is captured for the modified rows, along with the metadata needed to apply the changes to a target environment. In this process, the data is captured *asynchronously* (implemented with the help of SQL Agent jobs) by reading the database transaction log. Since it is an asynchronous process, it has a low impact on the performance of the system. This process is more efficient than the previous replication methodologies and it is particularly effective as we scale up on the database size.

CDC Implementation - Prerequisites

- SQL Server Agent Service should be running in order to get notified of the CDC actions
- Enable CDC for database by a member of the sysadmin fixed server role
- Enable CDC for individual table by the members of the db_owner fixed database role

CDC Implementation - Steps

The following steps will implement Change Data Capture (This feature has been implemented and tested with Microsoft SQL Server code name "Katmai" (CTP) - 10.0.1300.13 - Feb 2008)

1. Create a database with name 'TestDb' and enable CDC.

```
use master
GO
CREATE DATABASE TestDb
GO
use TestDb
GO
exec sys.sp_cdc_enable_db
```

```
GO
```

A new column **is_cdc_enabled** is added to sys.databases table. To check whether the CDC has been enabled for a database:

```
select is_cdc_enabled from sys.databases
where name = 'TestDb'
--1
GO
```

Once CDC has been enabled for a database, a new schema named **'cdc'** and a new user named **'cdc'** will be created. Also, following 5 new tables will be created (**Fig – 1**) -

- cdc.captured_columns - Contains one row for each column tracked in a capture instance. By default, all columns of the source table are captured. However, columns can be included or excluded when the source table is enabled for change data capture by specifying a column list.
- cdc.change_tables Contains one row for each change table in the database.
- cdc.ddl_history Contains schema modifications to source table
- cdc.index_columns Contains one row for each index column associated with a change table. The index columns are used by change data capture to uniquely identify rows in the source table. By default, the columns of the primary key of the source table are included. However, if a unique index on the source table is specified when change data capture is enabled on the source table, columns in that index are used instead. A primary key or unique index is required on the source table if net change tracking is enabled.
- cdc.lsn_time_mapping Contains one row for each transaction having rows in a change table. This table is used to map between log sequence number (LSN) commit values and the time the transaction committed. Entries may also be logged for which there are no change tables entries. This allows the table to record the completion of LSN processing in periods of low or no change activity.

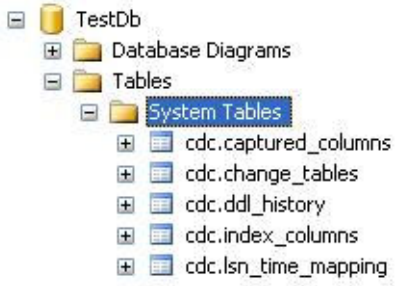

Fig 1: List of tables created in the database

1. Enable CDC for the table

Assume we have a table 'Employee'; enable CDC for the table, as below –

```
exec sys.sp_cdc_enable_table
@source_schema = 'dbo' --mandatory
, @source_name = 'Employee' --mandatory
, @role_name = 'cdc_test' --mandatory
, @supports_net_changes = 1
, @index_name = N'PK_Employee_EmployeeID'
, @captured_column_list = N'EmployeeID,EmpName, EmpCity'
, @filegroup_name = N'PRIMARY';
GO
```

@source_schema = Is the name of the schema to which the source table belongs. *source_schema* is **sysname**, with no default, and cannot be NULL.

@source_name = Is the name of the source table on which to enable change data capture. *source_name* is

sysname, with no default, and cannot be NULL.

source_name must exist in the *current* database. Tables in the **cdc** schema cannot be enabled for change data capture.

@role_name = Is the name of the database role used to gate access to change data. *role_name* is **sysname** and cannot be NULL. If the role currently exists, it is used. If the role does not exist, an attempt is made to create a database role with the specified name.

@supports_net_changes = Indicates whether support for querying for net changes is to be enabled for this capture instance.

supports_net_changes is **bit** with a default of 0. If 0 or if not specified, only support for querying for all changes is allowed. That is, only the system functions to query for all changes are created.

If 1, the system function to query for net changes is created in addition to the system function to query for all changes. If *supports_net_changes* is set to 1,*index_name* must be specified, or the source table must have a defined primary key. The columns defined in *index_name* or in the primary key are used to uniquely identify a row in the source table. These columns are added to the table cdc.index_columns.

@index_name = The name of a unique index to use instead of the primary key index to uniquely identify rows in the source table. *index_name* is **sysname** and can be NULL. *index_name* must be a valid unique index on the source table.

@captured_column_list = Identifies the source table columns that are to be included in the change table. *captured_column_list* is **nvarchar(max)** and can be NULL. If NULL, all columns are included in the change table. Column names must be valid columns in the source table. Columns defined in a primary key index, or columns defined in *index_name* must be included.

captured_column_list is a comma-separated list of column names. Individual column names within the list can be optionally quoted with either 'or []. If a column name contains an embedded comma, the column name must be quoted.

captured_column_list cannot contain the following reserved column names: **__$start_lsn**, **__$end_lsn**, **__$seqval**, **__$operation**, and **__$update_mask** as these are the metadata columns to allow interpretation of the change activity in the 'change table'.

captured_column_list cannot contain columns defined with a data type introduced in SQL Server 2008.

@capture_instance = Is the name that you assign to this particular CDC instance; you can have up two instances for a given table. (default:schema_sourcetable)

@filegroup_name = Is the filegroup to be used for the change table created for the capture instance. *filegroup_name* is **sysname** and can be NULL. *filegroup_name* must exist in the current database. If NULL, the default filegroup is used. A separate filegroup is recommended for creating change data capture change tables keeping the speed of data retrieval in the view.

To check whether the CDC has been enabled for a table:

selectis_tracked_by_cdc from sys.tables

where name = '<tablename>'

Note: If you make any schema changes to the table, you need to disable CDC at the table level and then re-enable it in order to capture changes to the data for the new columns.

To disable CDC on a table -

exec sys.sp_cdc_disable_table

@source_schema = 'dbo',

@source_name = 'Employee',

@capture_instance = 'dbo_Employee'

(or 'all', if you have more than one capture_instance name)

SQL Agent Jobs:

When the first table in a database has been enabled for change data capture, two new jobs are created and started by SQL Server Agent, as mentioned below"

- Job 'cdc.TestDb_capture'started successfully.
- Job 'cdc.TestDb_cleanup'started successfully.

A capture job is created using the default values when the first table in the database is enabled for change data capture and no transactional publications exist for the database. When a transactional publication exists, the transactional log reader is used to drive the capture mechanism, and this separate capture job is neither required nor allowed.

The cleanup job is created using the default values when the first table in the database is enabled for change data capture.

cdc.TestDb_capture job:

This job scans the database transaction log to pickup changes to the tables that have CDC enabled. The first step of the job is to raise an error event to the client indicating the start of the change data capture session using the T-SQL command RAISERROR(22801, 10, -1) and queries sys.messages for id = 22801. Once this is done, it executes the system stored procedure [sp_MScdc_capture_job] to start the Change Data Capture Collection Agent.

cdc.TestDb_cleanup job:

This SQL Server job purges the change tables periodically by executing the system stored procedure sys.sp_MScdc_cleanup_job.

Please note that the capture job should be kept running in order to capture changes to the source tables and the cleanup job is scheduled to run at regular intervals to remove the data from the change table. If the SQL Server agent or the capture job is stopped for any reason, all the changes that happened during this period will get captured from the transaction log when the job is started again.

Audit (Change Data) Table

When a table has CDC enabled, a new change table is created which corresponds to the main table. When the name of the change table is not specified at the time the source table is enabled, the table is named as: **cdc.*capture_instance*_CT** where *capture_instance* is the combination of schema name of the source table and the source table name in the format *schema_table*. (Ex: cdc.dbo_Employee_CT).

This table contains some book keeping columns such as **__$start_lsn**, **__$end_lsn (always populates with NULL; may be there for future use)**, **__$seqval**, **__$operation**, and **__$update_mask.**

The column __$operation has various codes as shown below –

Insert – 2

Update (row with before image) – 3

Update (row with after image) – 4

Delete – 1

Data consumers access change data through table-valued functions (**fn_cdc_get_all_changes_<capture_instance> & fn_cdc_get_net_changes_<capture_instance>**) rather than by querying the change tables directly.

In general, querying for **all** changes is more efficient than querying for **net** changes. If, for example, you know that the change data will consist entirely of inserts, a query for all changes using the **'all'** row filter option and a query for net changes using the 'all' or 'all with mask' option will return essentially the same resultset. The performance of the **cdc.fn_cdc_get_all_changes_<capture_instance>** function, however, is likely to be significantly better. If a primary key or unique index is not defined for a table, only queries using the **cdc.fn_cdc_get_all_changes_<capture_instance>** function are supported.

The function **cdc.fn_cdc_get_net_changes_<capture_instance>** returns only one change per row. If multiple changes are logged for the row during the request interval, the column values will reflect the final contents of the row.

Data is requested for changes having commit log sequence numbers (LSNs) that lie within a specified range. If a source row had multiple changes during the interval, each change is represented in the returned result set. In addition to returning the change data, four metadata columns provide the information you need to apply the changes to another data source. Row filtering options govern the content of the metadata columns as well as the rows returned in the result set. When the 'all' row filter option is specified, each change has exactly one row to identify the change. When the 'all update old' option is specified, update operations are represented as two rows: one containing the values of the captured columns before the update and another containing the values of the captured columns after the update.

Capture the change data based on time range:

```
DECLARE @begin_time datetime
DECLARE@end_time datetime
DECLARE@from_lsn binary(10)
DECLARE@to_lsn binary(10);
SET@begin_time = GETDATE()-1;
SET@end_time = GETDATE();
SELECT@from_lsn = sys.fn_cdc_map_time_to_lsn('smallest greater than or equal', @begin_time);
SELECT@to_lsn = sys.fn_cdc_map_time_to_lsn('largest less than or equal', @end_time);
/*Possible relational operators:
{ largest less than | largest less than or equal
| smallest greater than | smallest greater than or equal
}*/
select * from cdc.fn_cdc_get_all_changes_dbo_Employee(@from_lsn, @to_lsn, 'all');
```

Querying Configuration Information

You can return configuration information for a capture instance by using the stored procedure **sys.sp_cdc_help_change_data_capture**. The capture instance entry returned by the procedure identifies the columns of the source table that are being tracked and all capture instance configuration information.

	source_schema	source_table	capture_instance	start_lsn	end_lsn	supports_net_changes	role_name	index_name	filegroup_name
1	dbo	Employee	dbo_Employee	0x00000032000001020020	NULL	0	cdc_test	NULL	NULL

lsn	end_lsn	supports_net_changes	role_name	index_name	filegroup_name	create_date	index_column_list	captured_column_list
00000320000001020020	NULL	0	cdc_test	NULL	NULL	2008-04-25 18:01:22.163	NULL	[EmployeeID], [EmpName], [EmpCity]

Fig 2: CDC Configuration Information

Cleanup of change data: The data in the change table should be cleaned periodically as the table size grows very rapidly. The SQL Server cleans data in the change table for every 72 hrs which is configurable by the system administrator by updating the column **msdb.dbo.cdc_jobs.retention.** If you want the change data to be available for the analysis purposes even after 72 hours, we need to implement our own mechanism for archiving this information at regular intervals.

Conclusion

This paper discusses the implementation of Change Data Capture in SQL Server 2008 and how it is superior to the traditional change tracking mechanisms in the context of data warehousing applications.

About the Author: Suresh Babu Yaram has 10 years of experience in IT industry, worked extensively on various Microsoft Technologies in general & SQL Server database technology in particular. Currently, Suresh works with Satyam Computer Services as Database Architect.
