MATILLION

Why You Need Change Data Capture

FRICTIONLESS, REAL-TIME DATA INGESTION

What is Change Data Capture?

Change Data Capture (CDC) extracts data from relational databases and ingests it into a cloud data platform in near-real time. Once in the cloud, change data can be transformed and loaded into a target platform, like a cloud data warehouse, that is optimized for analytics. CDC is an excellent fit for time-sensitive decisions involving operational analytics, marketing personalization, recommendation engines, AI/ML modeling, and many more use cases.

Also, complete CDC replication should provide an immutable, historic copy of the raw data so that an exact replica of the original data can be recreated at any point in time. This comprehensive replication is often missed in many change data capture techniques. However, it is critical for database backup, disaster recovery, cloud migrations, fraud detection, and security or regulatory audits.

Why do you need Change Data Capture?

Relational databases store transactional records and are critical to operational business decisions. To understand what is happening, organizations analyze the data, but doing so within the database system often disrupts performance. To avoid disruptions, data is copied to a data warehouse that is optimized for queries and analytics. Traditionally, businesses used batch-based approaches to move data once or several times a day. However, batch ingestion itself can put a load on the operational tables and introduces latency (time delays), which reduces the operational value of data to modern organizations that need new data as soon as possible.

Extracting changes as they occur with CDC simplifies the replication process, is incredibly efficient, and consumes fewer resources. Data extracts with CDC are typically much smaller and have minimal, if any, impact on the performance of the database. Data is loaded into the cloud with extremely low latency–within seconds instead of hours or days–and maintains timely consistency across all downstream data stores and processing that rely on the data.

When should you use Change Data Capture?

Change Data Capture extracts and ingests data with extremely low latency. This speed is useful when replicating relational databases, migrating data from on-premises to cloud environments, and any analytics use cases where decisions need to be made in near-real time.

Here are some examples where low latency is critical to success:

Use case	Description	Business impact
Operational analytics	Ingests operational data in near-real time to help prepare it for analytics	Faster data ingestion leads to faster analytics, reporting, and decisions
Cloud migration	Performs initial load and syncs all changes in near-real time	Faster and more current replications
Fraud detection	All changes are captured to provide fraud detection algorithms with complete data.	Faster and more accurate detection of potential fraud
Real-time marketing campaigns	Timely and relevant personalization and next-action recommendations	Increases customer engagement, response rates, and high-value interactions
AI/ML	Trains models based on all current and historical data	More accurate models and reduced cycle times for model training
Synchronization	"Set and forget" data for simple or complex data integrations	Frees up time to focus on analyzing the data
Audits	Captures all changes (not just point-in-time snapshots) for full audit history	Recreates any state of the business at any point in time for full audit insight

What are common CDC methods?

There are several methods or techniques used for CDC, including timestamps, versioning, and status indicators on each database row. These are often referred to as "high watermarks" and are used to indicate the last replication or data capture. These methods take a point-in-time snapshot of the database and capture the changes since the last snapshot.

CDC Method	Implementation
Timestamp on rows	Each database row contains a column representing the time of the last change. The last timestamp for data capture is kept in a separate table. When data is captured, any row with a more recent timestamp than the last capture is extracted.
Version numbers on rows	Each database row contains a column indicating a version number that is updated each time a row changes. The last version of data capture is maintained in a separate table, When data is captured, any row with a higher version than the last capture is extracted.
Status indicators on rows	Each database row contains a status that is toggled to TRUE if that row changed. When data is captured, any row with a TRUE status is extracted, and then the status is reset to FALSE.
Time/Version/Status on rows	Combining the previous methods is a common practice and is not redundant or superfluous. Capturing change data can get very granular, but often becomes very complex. For example: "Capture all data for version 2.1 that changed between 5/1/2022 12:00am and 5/1/2022 3:00am where the status code indicates it is ready for production."

The problem with these methods is that if a data element changes multiple times between snapshots, only the last change is captured. The interim changes are completely missed and lost forever. This has huge implications for use cases that depend on analyzing every change event.

Change Data Capture can also be implemented through triggers or higher-level application programming.

CDC Method	Implementation
Triggers on tables	Each time data changes, a trigger logs the event in a separate shadow table. This method retains an immutable history of every change event, but it adds significant overhead, impacts the performance of the database, and complicates downstream query logic.
Event programming	Change Data Capture can be abstracted into the application layer. Here, application code can be written to indicate when data changes and copy that data into a separate database table. However, this creates extremely complex application code and brittle data pipelines.

The downside to both of these methods is that they can be complex, affect database performance, and require additional processing downstream to make the data useful.

What is the best CDC method?

The best method is **log-based CDC**, where **every** change appears as an entry in the database log files. Unlike point-in-time snapshot methods, log-based CDC captures every change event from the log files, not from the database operational tables. This method achieves the lowest latency, minimizes database resource consumption, and has little impact on the day-to-day operation of the database.

CDC Method	Implementation
Log-based scanners	Many database systems maintain a log record each time data or metadata is changed. By scanning and reading the change logs, this method captures every change event and has minimal, if any, impact on the performance of the database.

Log-based CDC is specific to a particular database management system. Most databases do not document the internal format of their transaction logs, although some – like PostgreSQL, Oracle, Microsoft SQL, MySQL, and DB2 – provide programming interfaces (APIs) to access their transaction logs.

Summary

Business transactions in relational databases are critical to understanding the state of business operations. Time-sensitive data replication and analysis is becoming more popular as businesses look to make more confident decisions faster.

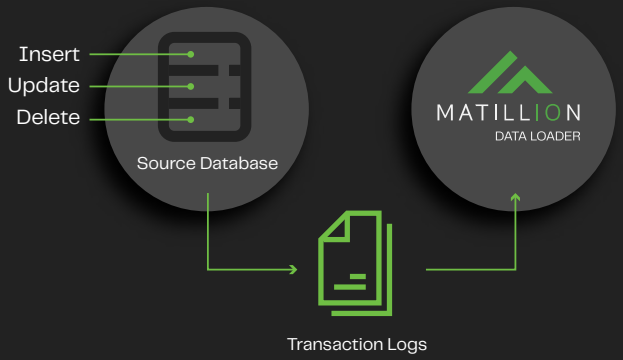
Change Data Capture, if done properly, can replicate data in near-real time and open up many new ways of working with data. However, the right CDC method must be employed. It is incumbent upon the data architect and data team to understand which CDC method works best for their specific use case, but if capturing every data change is critical, then log-based CDC is the best answer.

MATILLION

Matillion Data Loader with Change Data Capture

Matillion Data Loader with Change Data Capture leverages low-level database change logs, or log-based CDC, to ingest consistent, accurate, and up to date data. It captures every change, not just point-in-time snapshots that miss interim changes. Also, Matillion's CDC integrates with Matillion ETL to run data transformations – ensuring that data is not only up to date but also analytics ready in a minimal amount of time.

For more information, visit Matillion Data Loader or get started today.



Log-based Change Data Capture with Matillion Data Loader