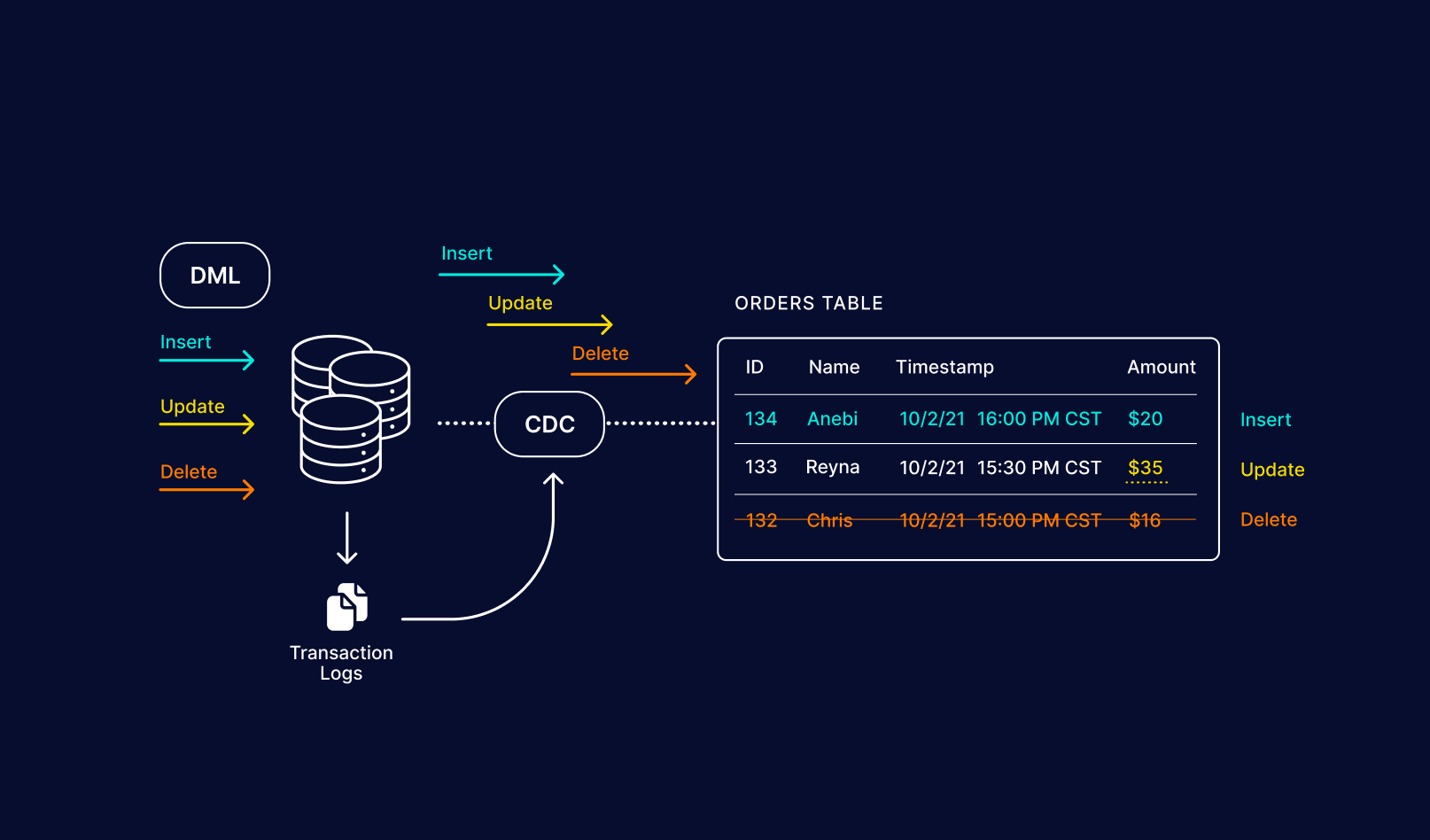
**Advantages of Log-Based CDC Over Trigger-Based CDC for Audits**

1. **Change Data Capture:** Change Data Capture (CDC) is a powerful feature in SQL Server that enables you to track changes made to your database. CDC allows you to capture insert, update, and delete operations performed on a SQL Server table and write them to a separate table. This can be useful for a variety of purposes, including auditing, replication, and data warehousing.

CDC uses a special type of table called a change table to store the captured changes. The change table has a similar structure to the original table but includes additional columns to track the type of change, the time of the change, and the transaction that caused the change.

Time is money when it comes to business. Change data capture is an ideal solution for companies looking to work with data more efficiently, as CDC works in real-time movement. After data collection in SQL Server, CDC helps move this information to a data warehouse, data lake, or other databases in real-time or near-real-time. The data movement efficiency that change data capture provides is extremely beneficial for organizations.

1. **Various methods used for CDC:** There are multiple common Change Data Capture methods that you can implement depending on your application requirements and tolerance for performance overhead.
   1. **Trigger-based CDC:** Traditionally, the most common technique used for capturing events was to use database or application-level triggers. The reason why this technique is still very widespread is due to its simplicity and familiarity. Triggers fire before or after INSERT, UPDATE, or DELETE commands (that indicate a change) and are used to create a change log.
   2. **Transaction log-based CDC:** Databases contain transaction logs (also called redo logs) that store all database events allowing for the database to be recovered in the event of a crash. With log-based change data capture, new database transactions – including inserts, updates, and deletes – are read from source databases’ native transaction logs.



In log-based CDC, the change data capture solution examines a database’s transaction log. During this process, the CDC solution reads the file to uncover the source system changes. This metadata information is stored in CDC change tables. Then, it executes data replication of these source changes to the target data store.

**Transaction Logs:** Every SQL Server database has a transaction log that records all transactions and the database modifications made by each transaction. The transaction log is a critical component of the database. If there's a system failure, you'll need that log to bring your database back to a consistent state.

1. **Comparative Analysis of Trigger-Based and Log-Based CDC:** Every RDBMS supports triggers, although with slightly different syntax and capabilities.In trigger-based CDC, the change data capture solution uses database triggers. During this process, the CDC solution runs when another event occurs. With the Trigger based approach we can implement CDC at the application level. CDC at the application level is defining database triggers and creating your own change log in shadow tables. Shadow tables can provide an immutable, detailed log of all transactions. However, they also add overhead to the source systems because they require a certain amount of run time each time the existing database refreshes. Significantly reduces the overall performance of the database by requiring multiple writes to a database every time a user inserts, updates, or deletes a row. Many application users do not want to risk the application behavior by introducing triggers to operational tables. DBAs and data engineers should always heavily test the performance of any triggers added into their environment and decide if they can tolerate the additional overhead.

Although the database or application-level triggers are a very common choice for CDC, there is a better way. The Audit Log is just a duplicate of the database transaction log (a.k.a redo log or Write-Ahead Log) which already stores row-based modifications. Therefore, you don’t really need to create a new Audit Log structure using database or application-level triggers, you just need to scan the transaction log and extract the CDC events from it. Log based CDC ensures high reliability with no missed changes, even if the system crashes or the network fails.

Parsing the transaction log of a database is complex, most databases do not document the format nor do they announce changes to it in new releases. This would potentially require you to change your database log parsing logic with each new database release. Additional log levels required to produce scannable transaction logs can add marginal performance overhead.

Each RDBMS used its own way of decoding the underlying transaction log -

* Oracle offers GoldenGate
* SQL Server offers built-in support for CDC
* MySQL, through various 3rd party solutions, like LinkedIn’s DataBus

1. **Advantages of Log based CDC and Trigger based CDC:**

**Trigger based CDC –**

* 1. Easy implementation.
  2. Shadow tables can provide a detailed log of all transactions.
  3. Receives direct support in the SQL API for some databases.
  4. Write custom logic.

**Log based CDC –**

1. Minimal Overhead.
2. High reliability with no missing changes.
3. No requirement to change the production database system’s schemas or the need to add additional tables.
   1. Can maintain ACID reliability across multiple systems.
   2. Log-based CDC is generally more scalable.
   3. Log-based CDC can handle high transaction volumes more efficiently.
4. **Disadvantages of Log based CDC and Trigger based CDC:**

**Trigger based CDC –**

* 1. Can experience trigger overload.
  2. Triggers may be disabled during certain operations.
  3. Significantly reduces the overall performance of the database by requiring multiple writes to a database every time a user inserts, updates, or deletes a row.
  4. Requires modifications to the database schema.
  5. If triggers are improperly configured we miss the changes.

**Log based CDC –**

* 1. Works only with databases that support log-based CDC.
  2. Parsing the transaction logs of a database is complex.
  3. Additional log levels required to produce scannable transaction logs can add marginal performance overhead.

1. **Benefits of log-based CDC over trigger-based CDC:**
   1. Performance: Log-based CDC captures changes directly from the transaction log, which minimizes performance overhead on the database operations. In contrast, triggers can add significant overhead, especially on high-transaction tables, because they execute additional logic with every insert, update, or delete operation.
   2. Scalability: Log-based CDC is generally more scalable. As the volume of transactions grows, triggers can become a bottleneck, whereas log-based CDC can handle high transaction volumes more efficiently.
   3. Ease of Management: Managing triggers can become cumbersome as the number of tables and triggers grows. Log-based CDC centralizes change capture and simplifies management.
   4. Minimal Risk of Data Corruption: Trigger-based CDC can sometimes lead to data corruption if there are errors in trigger logic. Log-based CDC minimizes this risk as it does not involve custom code execution.
   5. Reduced Development and Maintenance Costs: Implementing and maintaining triggers can be resource-intensive. Log-based CDC leverages existing SQL Server features, reducing the need for custom development and ongoing maintenance.
2. **Implementation of CDC in SQL Server:** Change data capture records, inserts, updates, and deletes activity that applies to an SQL Server table, which means organizations can capture changes in SQL Server data by using the SQL Server change data capture feature. However, the data system must meet certain prerequisites before you can enable CDC SQL Server. These prerequisites include:
   1. Having “sysadmin” privileges
   2. Running SQL Server Developer, Enterprise, or Standard Edition, as the web does not support CDC functionality
   3. Ensuring the SQL Server Agent runs on an SQL Server instance.

The user can use the following steps to implement SQL Server change data capture –

1. Open the SQL Server Management Studio and create a database.
2. Create a table.
3. Enable CDC on the database.
4. Define the specific table on which to enable change data capture.
5. Insert, Update or Delete the values into the table.
6. Verify that the change data capture is working.

Change Data Capture tracks the INSERT, UPDATE and DELETE operations on the database table, and records detailed information about these changes in a mirrored table, with the same columns structure of the source tables, and additional columns to record the description of these changes.

The additional columns include -

**\_\_$start\_lsn and \_\_$end\_lsn**: That show the commit log sequence number (LSN) assigned by the SQL Server Engine to the recorded change.

**\_\_$seqval:** That shows the order of that change related to other changes in the same transaction.

**\_\_$operation:** That shows the operation type of the change, where 1 = delete, 2 = insert, 3 = update (before change), and 4 = update (after change).

**\_\_$update\_mask**: That is a bit mask defined for each captured column, identifying the updating columns.

**SQL Commands:**

****

****

1. **References:** 
   1. [**https://www.integrate.io/blog/how-to-implement-change-data-capture-in-sql-server/**](https://www.integrate.io/blog/how-to-implement-change-data-capture-in-sql-server/)
   2. [**https://www.striim.com/blog/change-data-capture-cdc-what-it-is-and-how-it-works/**](https://www.striim.com/blog/change-data-capture-cdc-what-it-is-and-how-it-works/)
   3. [**https://vladmihalcea.com/a-beginners-guide-to-cdc-change-data-capture/**](https://vladmihalcea.com/a-beginners-guide-to-cdc-change-data-capture/)
   4. [**https://www.sqlshack.com/change-data-capture-for-auditing-sql-server/**](https://www.sqlshack.com/change-data-capture-for-auditing-sql-server/)
   5. [**https://learn.microsoft.com/en-us/sql/relational-databases/logs/the-transaction-log-sql-server?view=sql-server-ver16**](https://learn.microsoft.com/en-us/sql/relational-databases/logs/the-transaction-log-sql-server?view=sql-server-ver16)