Historical Data Preservation for a Complete View of an Organisation's Experience

Implementing CDC and SDC2 Across Data Platforms

July 2025

DL Lim

CONTENTS

1. EXECUTIVE SUMMARY

Failing to capture and preserve historical data is no longer an option. In industries like finance, insurance, and healthcare, organisations must be able to reproduce past results for compliance, audits, and analysis. Full database copies are costly and inefficient. Using Slowly Changing Dimensions (SCD Type 2) and Change Data Capture (CDC) allows companies to store only the meaningful changes while maintaining a complete historical record. This reduces storage needs, improves efficiency, and ensures data integrity over time.

The benefits go beyond compliance. Historical data provides context for long-term forecasting, root-cause analysis, and decision-making. By combining delta-based storage with cold and archival storage strategies, organisations can balance cost with accessibility. With standardised ingestion patterns and dedicated maintenance, companies gain trust in their data, unlock new insights, and turn their existing data into a competitive advantage.

2. INTRODUCTION

In certain industries like Finance and Insurance, the ability to reproduce results based on a historical snapshot of data is of paramount importance. This is often a requirement for regulatory compliance.

In insurance for instance, many models, sometimes deterministic and sometimes stochastic, make projections decades into the future, based on a snapshot of the currently available policyholder data, data from the Bank of England, Bloomerg and various other sources.

Each time such a model runs, an audit trail is required, so that if the same model had to be run 5 years later in case of an ad hoc discrepancy analysis, they are able to perform the same run and reproduce the same results from the same set of data.

We'd be fortunate if that same data even still exists five years later, let alone in the exact form it once did! The reality is that third-party data sources evolve, update, and sometimes even retire entire datasets. Even high-trust authoritative providers such as the Bank of England may present slightly different figures for the same timeframe as methodologies, revisions, or corrections are introduced over the years.

This natural drift is unavoidable, and so it falls on the organisation to ensure its own retention strategy is robust enough to preserve a reliable historical record. In practice, this means treating snapshots as frozen moments in time, faithfully captured, carefully stored, and ready to be re-examined in the future with full confidence in their integrity.

3. PROBLEM STATEMENT

3.1 HOW WE STORE "EVERYTHING"

An often-used quick solution for storing a copy of operational data on an analytical platform is to use a simple full refresh. This is just a lagging copy of the source, and does not store any history.

If an organisation has deep pockets and storage (and compute) is cheap, store everything! Full loads and complete ingestion each pull. This means complete copies of the database upon each ingestion interval. 12 copies over a year would mean approximately 12 times the size of the original data on average!

However, one cannot conscionably endorse such practices when many better solutions exist, especially those with a better environmental and social impact. (more space used = more footprint)

We can choose to use cleverer ways of storing data efficiently.

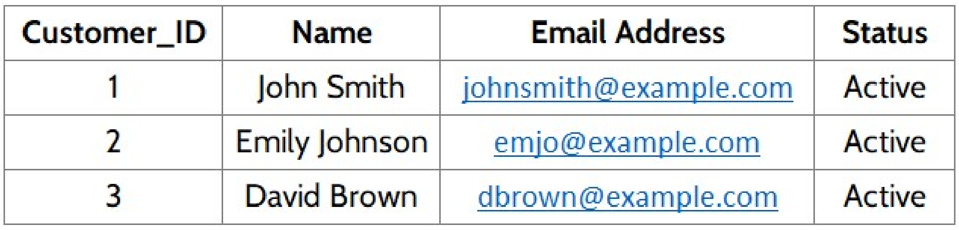
3.2 PRACTICAL SITUATIONS

Consider two simple situations which may be commonly faced by teams and organisations around the world.

3.2.1. A COMMON SITUATION #1

Denise was tasked with extracting customers emails from the CRM to be used in their email marketing campaigns. However, the CRM only exports a csv with the full list of customer information.

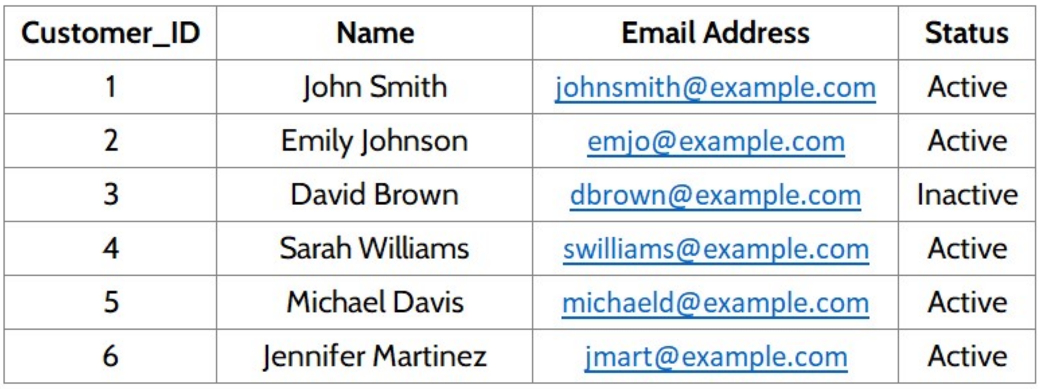
Here's the list when she extracted it on 2nd Feb. (For argument's sake, this list is 20MB in the database)



Customer_ID	Name	Email Address	Status
1	John Smith	johnsmith@example.com	Active
2	Emily Johnson	emjo@example.com	Active
3	David Brown	dbrown@example.com	Active

(To keen-eyed readers, yes, we're aware of PII and we're overlooking that for the sake of this example!)

She ran a campaign with that table, and it was successful. Seeing the returns on this, her manager asked if she could do the same again in March. And on 5th March she got this data from the CRM, occupying 70MB on the disk:



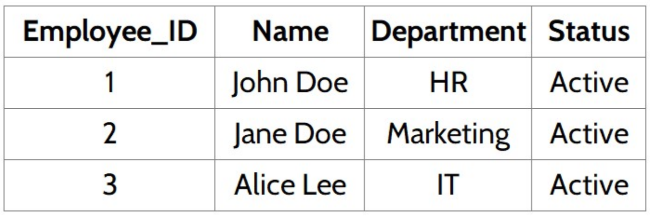
Customer_ID	Name	Email Address	Status
1	John Smith	johnsmith@example.com	Active
2	Emily Johnson	emjo@example.com	Active
3	David Brown	dbrown@example.com	Inactive
4	Sarah Williams	swilliams@example.com	Active
5	Michael Davis	michaeld@example.com	Active
6	Jennifer Martinez	jmart@example.com	Active

Because she didn't have a policy on change data capture, she thought it would be ok to just store both sets of data as different tables in the database. This means that there is now 90MB of data, with some duplicate rows, and no additional information gained from storing duplicates.

One can imagine what is going to happen as she continues her marketing campaigns in the following months as the company grows. The storage requirement is going to skyrocket exponentially.

3.2.2. A COMMON SITUATION #2

Let's consider another toy example. This table lists employees at a company, as at 3rd March.

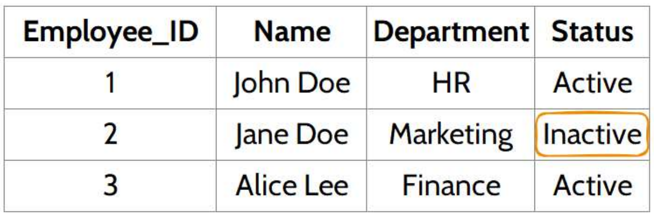


Employee_ID	Name	Department	Status
1	John Doe	HR	Active
2	Jane Doe	Marketing	Active
3	Alice Lee	IT	Active

On 4th April, Jane Doe left the company.

On 8th April, Alice Lee transferred to the Finance department.

This is how the new table looks like



Employee_ID	Name	Department	Status
1	John Doe	HR	Active
2	Jane Doe	Marketing	Inactive
3	Alice Lee	Finance	Active

Now, we fast forward 6 months in to the future - it is now 10th October.

Alice asks the HR owner of the table, John Doe - "Hey, do you remember when Jane left the company? I'm trying to allocate expenses for her time with us"

John Doe says "I don't remember clearly, but I think it was a couple of days before you transferred over to Finance".

Alice says, "Ah, I was going through many things at the time, and I can't remember when it was exactly. Let's pen it down as 15th April then"

ⓘ Note

Naturally, this situation can easily be remedied if we had kept a clear, automated log of each "transaction" or change in the database

4. PROPOSED SOLUTION

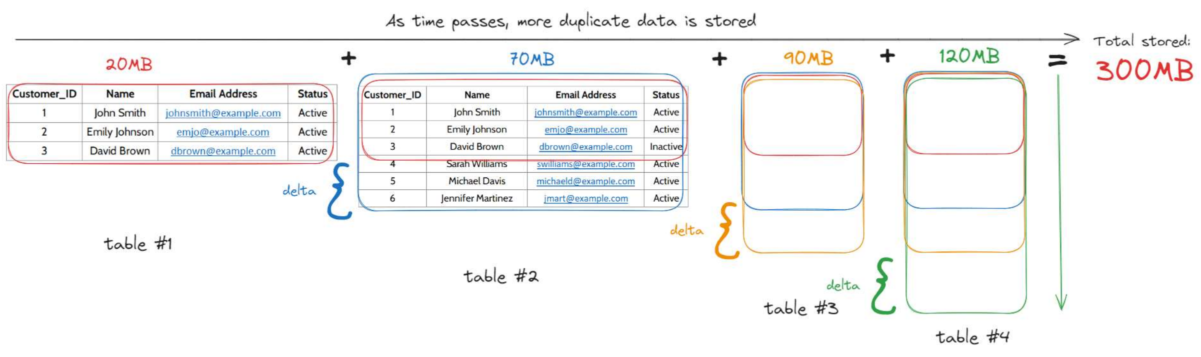
There's a concept called Slowly Changing Dimensions (SCD), and with it, there are several types of SCD.

What we have seen in the second example is SCD Type 1 - where any new values received will overwrite the existing entries.

A more powerful approach is to use SCD Type 2 - where each difference, also called the delta, is captured in a process known as Change Data Capture (CDC).

If we consider the first example, we can observe that the data storage requirements will increase exponentially as the data increases, but there is duplicate information that isn't necessary to be stored. If we chose only to capture the differences between each pull, we can form a new table using just the delta's.

Full Snapshot Load

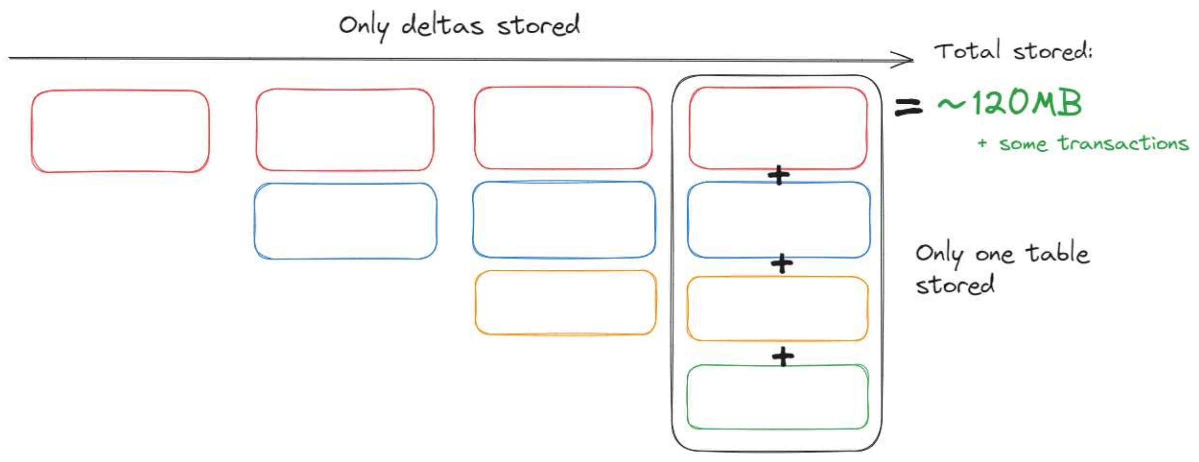


Customer_ID	Name	Email Address	Status
1	John Smith	johnsmith@example.com	Active
2	Emily Johnson	emjo@example.com	Active
3	David Brown	dbrown@example.com	Active

Customer_ID	Name	Email Address	Status
1	John Smith	johnsmith@example.com	Active
2	Emily Johnson	emjo@example.com	Active
3	David Brown	dbrown@example.com	Inactive
4	Sarah Williams	swilliams@example.com	Active
5	Michael Davis	michaeld@example.com	Active
6	Jennifer Martinez	jmart@example.com	Active

By storing only the delta, we maintain only one table, and no duplicates are kept.

SCD Type 2



"But wait! From the first to the second table, David Brown went from Active to Inactive! What do we do with that?"

Good catch! That's what the second example aims to illustrate. A record of any changes in existing data can also be retained.

Given the original data, let's add two metadata columns Valid_From and Valid_To. In this example, we'll assume NULL means it goes up to the year 9999, and thus is still currently valid.

Employee_ID	Name	Department	Status	Valid_From	Valid_To
1	John Doe	HR	Active	03-Mar	NULL
2	Jane Doe	Marketing	Active	03-Mar	NULL
3	Alice Lee	IT	Active	03-Mar	NULL

Now that we have two new columns, we know two changes occured in April. When we input these two entries, we also change the "Valid_To" for a given matching "Employee_ID". Instead of overwriting the original entries, we add them as new entries.

Here's what it looks like:

Employee_ID	Name	Department	Status	Valid_From	Valid_To
1	John Doe	HR	Active	03-Mar	NULL
2	Jane Doe	Marketing	Active	03-Mar	04-Apr
3	Alice Lee	IT	Active	03-Mar	08-Apr
2	Jane Doe	Marketing	Inactive	04-Apr	NULL
3	Alice Lee	Finance	Active	08-Apr	NULL

Note that we've not changed the original entries other than the validity period, but we have now added two more new rows to show the most current record.

Let's say we want to find out what Jane Doe was up to on 26 March, we can do a

```
SELECT * FROM table
WHERE '26-Mar' BETWEEN Valid_From AND Valid_To;
```

Employee_ID	Name	Department	Status	Valid_From	Valid_To
2	Jane Doe	Marketing	Active	03-Mar	04-Apr

Note: Date format for query is for illustration purposes only.

Note that the Employee_ID is no longer the primary key here, since Employee_ID can be repeated multiple times as more changes are made to the same employee.

The way a table like this identifies the unique key is through a composite key of Employee_ID, Valid_From and Valid_To.

And thus, we have now implemented CDC using SCD Type 2.

While not all organisations require data to be mission-critical nor do they need the historical versioning, it is often a good practice to ensure the reproducibility and the integrity of data by preserving the changes made over the lifetime of the datasets.

5. BENEFITS/IMPACT

By implementing CDC in most, if not all data platform transactions, an organisation could theoretically capture everything that has ever been experienced by the organisation, stored securely even if the sources go down, retrievable at any time for whichever desired time range.

5.1 REGULATORY AND COMPLIANCE ASSURANCE

CDC together provide a verifiable audit trail of data as it changes over time. This ensures that historical states of data can always be reproduced, which is critical for meeting the stringent requirements of regulators in finance, insurance, healthcare, and other highly regulated sectors.

5.2 ACCURATE HISTORICAL CONTEXT

SCD2 ensures that data changes are not overwritten but instead preserved with a timeline. This means decision-makers can see not only the current state of data but also how it has evolved. For long-term models, forecasting, or root-cause analysis, this historical fidelity is invaluable.

Historical data retention allows businesses to “rewind” and re-run models or simulations as if they were happening at the original point in time.

5.3 IMPROVED DATA QUALITY AND TRUST

By capturing every change through CDC and storing historical versions with SCD2, organisations avoid the risks of silent data loss or untraceable updates. Stakeholders gain confidence that the data reflects reality both now and at any point in the past.

As trust on a platform increases, the barriers to obtaining business value from data are lowered. More users can have faith that even if (as an example) a faulty legacy system goes down due to political changes or otherwise, they will still be able to obtain its data from the data platform.

6. CHALLENGES AND CONSIDERATIONS

6.1 NOT ALL SOURCES ARE CREATED EQUALLY

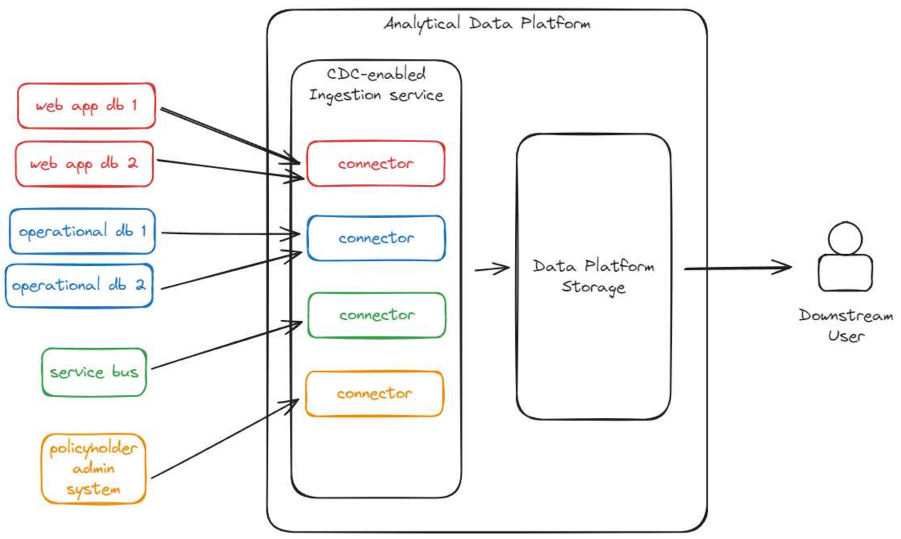
Depending on the data source, ingestion pipelines must be designed differently. Structured databases, NoSQL stores, and service bus or event-driven architectures all require distinct approaches to ensure reliable data flow. The choice of pipeline depends on the type of data, its volume, and the format in which it is generated.

The frequency of ingestion and ongoing maintenance are critical considerations. Some sources require near real-time updates, while others can be ingested in batches. Maintaining pipelines also varies in complexity depending on the source's stability, schema changes, and the level of transformation needed.

Delta, tracking changes efficiently, is another important factor. For operational systems, extracting only the changes reduces load and keeps the data platform up to date without overwhelming it. This is particularly relevant for NoSQL or event-driven sources, where data streams may be continuous and high-volume.

Understanding the source of truth is essential. Operational databases are often the most accurate for transactional data, whereas analytical workloads can rely on ingested source data on the platform. However, the required granularity and speed of access will determine whether real-time ingestion is necessary or if batch updates are sufficient. Not every workload demands instant data, and designing pipelines accordingly prevents unnecessary complexity.

Source Technologies May Vary



6.2 THE COST OF "EVERYTHING"

Storing every single record in full is expensive and often unnecessary. Duplicates don't provide any additional information, but takes up physical space in storage. By processing only deltas instead of full loads, we only need to store a multiple of the number of refreshes, which can reduce storage costs significantly. This approach also lowers ETL processing time, particularly for high-volume data.

Timestamps, which are enabled by using SCD2, are crucial for managing storage efficiently. Data can be moved into cold storage after an elapsed period, such as 1–2 years (or depending on frequency of access), helping reduce costs for less frequently accessed information.

Longer-term retention can be managed by parking older data into archival storage, with organisational policies and regulatory requirements guiding whether data is deleted after a set period, if storage costs become unmanageable. For example, the UK's Financial Conduct Authority (FCA) sets 7 years on a number of record types, amongst other retention periods per their retention schedule.

However, it is important to recognise the overhead of implementing CDC everywhere. While CDC can provide near real-time updates, the complexity and maintenance burden can outweigh the benefits for some sources. Organisations need to balance cost, effort, and the value of up-to-date information when deciding where to deploy it.

Some of that implementation cost can be offset by having a dedicated team that maintains a set of libraries for such ingestion patterns across the org. With coherence and homogeneity in these integration patterns, maintenance cost becomes manageable, while reaping all the benefits of SCD2.

6.3 ADDRESSING SCHEMA DRIFT OVER TIME

Just as data sources can disappear over time, a less disruptive effect commonly observed is that the structure of source data may change. This refers to the addition/deletion/renaming of columns at source, which often affect downstream consumers.

Depending on pre-existing data contracts, it may not always be conducive to propagate upstream changes to downstream consumers. Transformation layers of data abstraction may exist between producers and consumers.

Changes may be categorised as planned or unplanned, and robust sign-off processes should be put in place to handle every possible scenario to minimise downtime.

The addition of columns in the implementation may also lend itself to building a catalog of rich, structured metadata that lends itself nicely to the provision of centralised metadata management and governance.

Teams no longer need to build complex custom pipelines just to maintain historical accuracy; instead, they leverage the platform’s built-in snapshotting, lineage tracking, and access controls.

In practical terms, this reduces operational overhead, improves confidence in historical reporting, and accelerates regulatory audits or model re-runs.

7. CONCLUSION

The benefits of implementing Change Data Capture and Slowly Changing Dimensions (Type 2) is to have a full historical record of an organisations experience such that a privileged user may time travel to any point in the history of the organisation and have a complete picture of their data state as at that point in time.

For some, this may be a compliance exercise in meeting regulatory requirements, though all of this could well be reframed as a forward-looking initiative that provides a full backwards view, pun intended.

By seeking opportunistic advantages in data, an organisation can gain a new competitive edge from data that they already have and use on a daily basis.

8. REFERENCES

- https://aws.amazon.com/blogs/big-data/implement-historical-record-lookup-and-slowly-changing-dimensions-type-2-using-apache-iceberg/
- https://support.sas.com/resources/papers/proceedings16/7560-2016.pdf
- Kimball, Ralph; Ross, Margy (July 1, 2013). The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3rd Edition. John Wiley & Sons, Inc. p. 122. ISBN 978-1-118-53080-1.
- Ross, Margy; Kimball, Ralph (March 1, 2005). "Slowly Changing Dimensions Are Not Always as Easy as 1, 2, 3". Intelligent Enterprise.