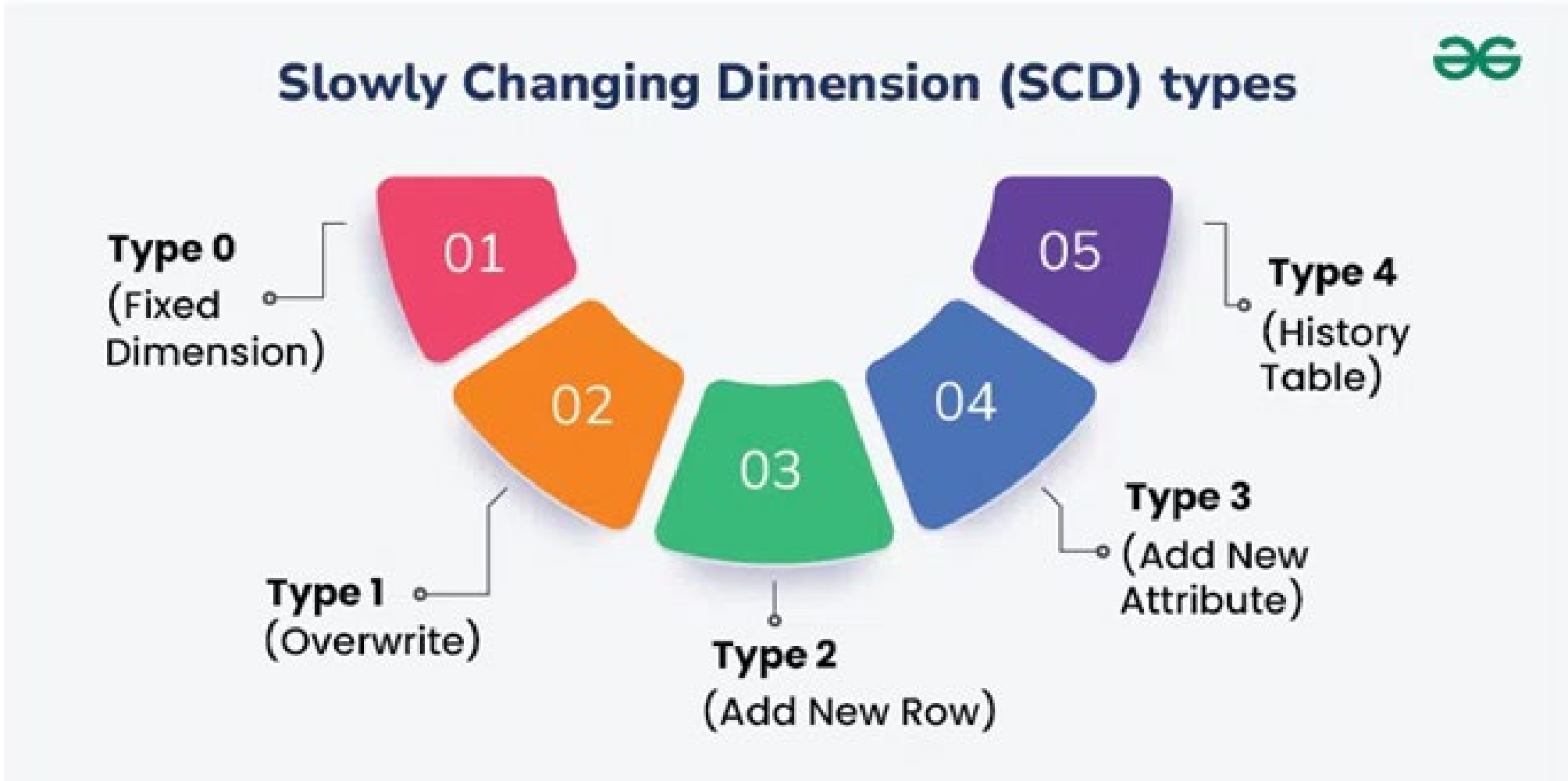
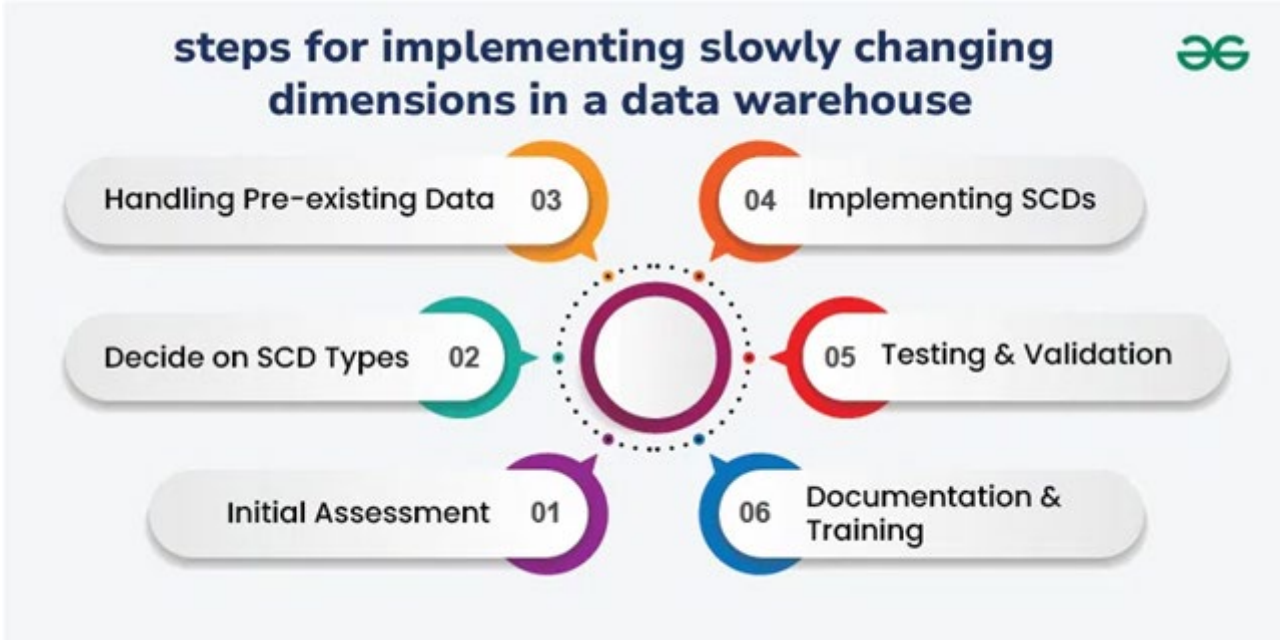
Data Engineer

SCD:

alfatraining




steps for implementing slowly changing dimensions in a data warehouse
Handling Pre-existing Data 03
04 Implementing SCDs
Decide on SCD Types 02
05 Testing & Validation
Initial Assessment 01
06 Documentation & Training

Definition

A Slowly Changing Dimension (SCD) is a dimension that stores and manages both current and historical data over time in a data warehouse. It is considered and implemented as one of the most critical ETL tasks in tracking the history of dimension records.

SCD 0

SCD Type 0: Retain Original

Type 0, also known as **Retain Original** type, refers to dimensions or attributes that never change and will not be updated in the data warehouse. The original dimension value is always retained, and no tracking of historical data takes place. Type 0 is applicable to most date dimension attributes.

Examples of SCD Type 0 include:

- Date of birth
- An employee's start date with a company
- Social security number

SCD 1

The Slowly Changing Dimension transformation supports four types of changes: changing attribute, historical attribute, fixed attribute, and inferred member. Changing attribute changes overwrite existing records. This kind of change is equivalent to a Type 1 change.

SCD Type 1: Overwrite

Type 1, or **Overwrite** type, refers to an instance where old data is overwritten with new data. The old data is lost as it is not stored anywhere, and the latest snapshot of a record is maintained in the data warehouse without any historical records. With this SCD type, you cannot keep track of changes over time.

This is the default type of dimension you create; you don't need to specify additional information to create an SCD Type 1.

The benefits of SCD Type 1 include requiring less storage space and being easy to maintain. However, the disadvantage of the Type 1 method is that historical data is lost. Also, it doesn't support the analysis and reporting of historical data.

This type is commonly used to correct errors in a dimension, updating values that were irrelevant or wrong. No history is retained.

Here are some SCD Type 1 examples:

- **Customer address data**: The old address is replaced with the new address, and the history of the old address is not required.
- **Inventory data**: When the stock of an item is updated, the new stock level replaces the old stock level.
- **Employee salary data**: When an employee's salary is updated, the new salary replaces the old salary, and the history of the old salary is not maintained.

Type 1: Update Changes

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State
123	ABC	Acme Supply Co	CA

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State
123	ABC	Acme Supply Co	IL

SCD 1 + 2

Type 1 Slowly Changing Dimension: This method overwrites the existing value with the new value and does not retain history.

Type 2 Slowly Changing Dimension: This method adds a new row for the new value and maintains the existing row for historical and reporting purposes.

SCD Type 2: Add New Row

SCD Type 2, also known as the **historical tracking method**, helps maintain the history of changes in a dimension table. Every time there is a change in the source system, a new row is added to the dimension table with a unique identifier. The resulting table will retain the prior history, allowing full history tracking.

There are two ways to handle SCD Type 2 dimensions:

- Adding a **flag column** signifying the record that is currently active. The flag column uses a simple indicator (such as 0/1 or Y/N) to denote whether a specific record is currently active or a valid version.
- Adding one or two **timestamp columns** to signify when a new record was created or made active and when it was made ineffective.

The benefit of SCD Type 2 is that no data is lost as the history of a record is maintained. This helps with accurately tracking changes over time. However, it can be complex to implement and requires more storage space. Adding new rows to the table adds to the size, eventually making the table unmanageable.

This method is suitable for situations where you require historical data for reporting and analysis. But, if there is a possibility that the structure of the data will change (such as new columns being added to the table), this type is less likely to be used.

Examples of SCD Type 2 include:

- **Sales data**: You can track your sales of a particular item or profit over time to help you analyze your sales.
- **Customer or order data**: You can keep track of previous orders of a person and recommend items based on that.
- **Customer address**: When a customer relocates, a new address entry with a new timestamp is added to the data warehouse. The previous address is retained with its respective timestamp.

Type 2: Keep Historical

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Start_Date	End_Date
123	ABC	Acme Supply Co	CA	01-Jan-2000	21-Dec-2004
124	ABC	Acme Supply Co	IL	22-Dec-2004	

Type 2 Slowly Changing Dimension

Product Dim (Source)

Product Name	Product ID	Product Descr
12 inch box	012	12 inch glued box
10 inch box	010	10 inch ~~glued~~ box 10 inch **pasted** box

Product Dim (Target)

SID	Source Product ID	Product Name	Product Descr	EFF_ START_DT	EFF_ END_DT
0001	012	12 inch box	12 inch glued box	Jan-01-1753	Dec-31-9999
0002	010	10 inch box	10 inch glued box	Jan-01-1753	May-12-06
0003	**010**	**10 inch box**	**10 inch pasted box**	**May-12-06**	**Dec-31-9999**

… SCD 3

Type 1 – This model involves overwriting the old current value with the new current value. No history is maintained.
Type 2 – The current and the historical records are kept and maintained in the same file or table.
Type 3 – The current data and historical data are kept in the same record.

SCD Type 3: Add New Attribute

Similar to SCD Type 2, you can also track record changes with SCD Type 3. However, instead of adding a new row for every change, a new column is added to track the previous value. But, this method preserves the most recent history, as it is limited to the number of columns designated for storing historical data.

The SCD Type 3 stores two versions of values for the selected level attributes. Each record will store the current and previous values of the selected attribute. Whenever the value of the attribute changes, the current value gets stored as the old value, and the new value becomes the current value.

You can only track one change in a record rather than multiple changes over time. This method isn't scalable if you want to preserve complete history, as it allows only to keep the latest version of the history.

The benefit of this SCD type is it requires less storage space. Also, it enables fast queries since there's limited history to scan. However, it doesn't provide a complete history of changes since it only tracks limited changes. For columns that require full history, it might not be suitable for tracking changes.

An example of SCD Type 3 is in **product pricing** or **financial reporting**, where the previous and current values of a service or product are important.

Type 1: Update Changes

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State
123	ABC	Acme Supply Co	CA

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State
123	ABC	Acme Supply Co	IL

Type 2: Keep Historical

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Start_Date	End_Date
123	ABC	Acme Supply Co	CA	01-Jan-2000	21-Dec-2004
124	ABC	Acme Supply Co	IL	22-Dec-2004	

Type 3: Preserve Limited History

Supplier_Key	Supplier_Code	Supplier_Name	Original_Supplier_State	Effective_Date	Current_Supplier_State
123	ABC	Acme Supply Co	CA	22-Dec-2004	IL

SCD 4

Slowly changing dimension type 4 is used when a group of attributes in a dimension rapidly changes and is split off to a mini–dimension. This situation is sometimes called a rapidly changing monster dimension. An example of SCD Type 4 is when a product's price changes. The old price and the effective date are added to a separate history table, while the current price is stored in the main dimension table. Slowly changing dimension *type 4* is used when a

group of attributes in a dimension rapidly changes and is split off to a *mini–dimension*. This situation is sometimes called a ***rapidly changing monster*** *dimension*. Frequently used attributes in multimillion-row dimension tables are mini-dimension design candidates, even if they don’t frequently change. The type 4 mini-dimension requires its own unique primary key; the primary keys of both the base dimension and mini-dimension are captured in the associated fact tables.

SCD Type 4: Add History Table
This slowly changing dimension involves maintaining the records in two different tables—a **current record table** and a **historical record table**. Changes are tracked in a separate history table. While the main dimension table stays current, the history table stores past data.
With SCD Type 4, a record will be added to the history table for each change in the source system. It's useful for keeping track of records that have many changes over time. It allows to retain both the original and updated data while still maintaining a small footprint for the main table. The method resembles how change data capture techniques and database audit tables function.
An example of SCD Type 4 is when a product's price changes. The old price and the effective date are added to a separate history table, while the current price is stored in the main dimension table.

SCD Type 4 current table

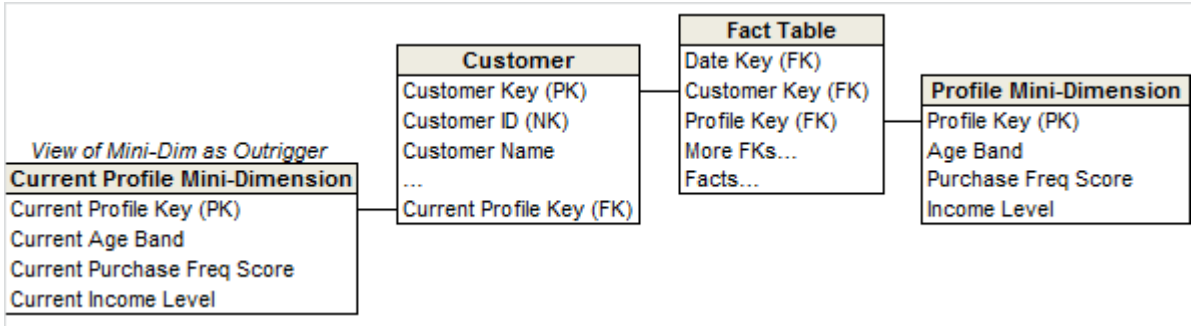
cust_id	customer_number	first_name	last_name
1	10001	Context	Wonderful

SCD Type 4 history table

cust_id	customer_number	first_name	last_name	effective_date
1	10001	Kontext	Wonderful	2022-01-01
1	10001	Context	Wonderful	2022-07-01

SCD 5

The reason why this is called type 5? Because 4+1=5. This is a combination of these two types, and it entails embedding a current profile mini-dimension key in the base dimension table that gets overwritten as a type 1 attribute. Both the dimension and mini-dimension profiles are shown in a single table.



SCD 6

SCD Type 6: Combined Approach
The SCD Type 6 is a **hybrid approach** of Type 1, Type 2, and Type 3 (**1 + 2 + 3 = 6**). This includes columns for both historical and current data and a column to track the current version of the record.
Both historical and current data can be stored in the same row, with the current version being easily accessible. This SCD type is useful when a business wants to view both current and historical data in the same report.
An example of SCD Type 6 is when an employee's role and department might change. While the role change can be presented in a new row (Type 2), the department can be overwritten (Type 1). A column can indicate the previous role (Type 3).

Original row in Product dimension:

Product Key	SKU (NK)	Product Description	Historic Department Name	Current Department Name	Row Effective Date	Row Expiration Date	Current Row Indicator
12345	ABC922-Z	IntelliKidz	Education	Education	2012-01-01	9999-12-31	Current

Rows in Product dimension following first department reassignment:

Product Key	SKU (NK)	Product Description	Historic Department Name	Current Department Name	Row Effective Date	Row Expiration Date	Current Row Indicator
12345	ABC922-Z	IntelliKidz	Education	Strategy	2012-01-01	2012-12-31	Expired
25984	ABC922-Z	IntelliKidz	Strategy	Strategy	2013-01-01	9999-12-31	Current

Rows in Product dimension following second department reassignment:

Product Key	SKU (NK)	Product Description	Historic Department Name	Current Department Name	Row Effective Date	Row Expiration Date	Current Row Indicator
12345	ABC922-Z	IntelliKidz	Education	Critical Thinking	2012-01-01	2012-12-31	Expired
25984	ABC922-Z	IntelliKidz	Strategy	Critical Thinking	2013-01-01	2013-02-03	Expired
31726	ABC922-Z	IntelliKidz	Critical Thinking	Critical Thinking	2013-02-04	9999-12-31	Current

Limitations of Slowly Changing Dimensions in Data Science

While SCDs are an essential part of data warehousing, allowing historical data to be preserved along with changes over time, there are some associated limitations, such as:

- **Storage Space:** A dataset with a vast number of input features can rapidly increase storage requirements. This is especially true for SCD Type 2, which involves adding a new row for each change.
- **Maintenance:** With an increase in the number of attributes or dimensions, tracking becomes complex. This can result in an increased possibility of errors.
- **Performance:** As the data volume increases to track historical changes, it may sometimes deteriorate the performance of the machine learning model.
- **Scalability:** With growing data and accumulated changes, some SCD types might not scale efficiently. This affects storage, computation, and visualization.

Solve Limitiations – Dimension reduction

Dimension reduction is a machine learning or statistical technique that reduces the number of features in a dataset while retaining as much important information as possible. Here, dimensions refer to the number of input features, variables, or columns of a given dataset.

•Slowly changing dimensions in data science involve the addition of new rows, columns, or attributes to capture changes, inherently adding to the dimensionality of the dataset. If you intend to build predictive models that use this historical data, multiple dimensions may result in overfitting. However, with dimension reduction, such datasets become more manageable and help avoid overfitting of models.

•With slowly changing dimensions in data science, every change captured results in an increase in data storage requirements. Additionally, processing datasets with increasing dimensions can be computationally expensive. However, dimension reduction can help overcome these challenges while ensuring that crucial information is retained.

•Most slowly changing dimensions in data science involve redundancy. For example, in SCD Type 2, a new row is added for every change in a dimension. Eventually, this results in multiple rows for the same entity and increased dimensionality. Dimension reduction helps identify and manage such redundancy.

Imlement/develop SCD

Ideally, slowly changing dimensions are best considered right at the start of your database creation. If you're just beginning to build your data infrastructure or working for a startup, this is easier to implement.

For existing databases, start by assessing the data that currently exists in your database. Document the different types of dimensions you find and how they relate to one another. If there are dimensions of type 0 that shouldn't be, start with those.

And if required, add historical tracking as soon as possible.

Next, determine whether you would prefer types 2, 3, or 4 and if they are suitable for your business. Some questions to evaluate are:

•How often do these dimensions change?
•Would you use a timestamp column or a flag column?
•Should you add a historical table that can store a lot of records?

Such a process will involve analytics engineers, data engineers, and data analysts.
Another important thing to consider is how you want to handle previous records that weren't tracked over time. While you can forgo the historical data and implement SCD for future operations, it would be best to put together old records and create your own version of a timestamp or flag column.

Links

https://learn.microsoft.com/en-us/sql/integration-services/data-flow/transformations/slowly-changing-dimension-transformation?view=sql-server-ver16
https://docs.oracle.com/cd/E41507_01/epm91pbr3/eng/epm/phcw/concept_UnderstandingSlowlyChangingDimensions-405719.html#:~:text=Type%201%20Slowly%20Changing%20Dimension%3A%20This%20method%20overwrites%20the%20existing,for%20historical%20and%20reporting%20purposes.
https://www.iri.com/blog/vldb-operations/introduction-to-slowly-changing-dimensions-scd/#:~:text=Type%201%20%E2%80%93%20This%20model%20involves,kept%20in%20the%20same%20record.

https://de.wikipedia.org/wiki/Slowly_Changing_Dimensions
https://onlinedatawarehouse.de/datenmodellierung/eine-einfuehrung-in-slowly-changing-dimensions-scd/
https://www.expressanalytics.com/blog/what-is-a-slowly-changing-dimension-and-the-logic-in-implementation/