ProjectPro

Learn How to Implement SCD in Talend to Capture Data Changes

CookBook

Table of Content:

Introduction

Capturing data changes is the process of monitoring and recording any modifications made to a dataset over time. This can be accomplished using various methods, including manual observation, trigger-based systems, audit trails, and change data capture (CDC).

1. **Manual observation**: This approach involves manually reviewing a dataset periodically and noting any changes that have occurred since the last review. This approach can be time-consuming, but it can be useful for small datasets or in situations where data changes occur infrequently.
2. **Trigger-based systems:** Trigger-based systems are designed to trigger an alert or notification when a change is made to a dataset. For example, a trigger could be set up to send an email notification when a new record is added to a database. This approach is automated and can be useful for datasets that change frequently.
3. **Audit trails**: Audit trails are a chronological record of all changes made to a dataset. They can be used to identify who made the change, when it was made, and what the change was. This approach provides a detailed history of changes to a dataset and can be useful for compliance or troubleshooting purposes.
4. **Change data capture (CDC):** CDC is a technology that captures changes to data in real-time as they occur. This approach is commonly used in databases and data warehouses to ensure that changes to data are accurately captured and stored. CDC can be used to replicate data from one database to another or to capture changes to a single database.

Overall, capturing data changes is important for maintaining data integrity, compliance, performance optimization, decision-making, and data governance. The method used for capturing data changes will depend on the specific requirements of the organization and the characteristics of the dataset being monitored. By accurately capturing data changes, organizations can ensure that they have access to current and accurate data, which is essential for effective decision-making and operations.

Capturing data changes is important for several reasons:

1. **Data quality**: When data changes occur, it's essential to capture those changes accurately to maintain the quality of the data. By capturing data changes, it's possible to ensure that the data is up-to-date, accurate, and complete.
2. **Compliance**: Many industries and government regulations require that data be captured and stored securely and that changes be tracked. For example, healthcare and financial institutions must comply with regulations that require them to maintain an audit trail of all changes to patient or financial data.
3. **Performance optimization:** Capturing data changes can help organizations optimize their performance by identifying patterns and trends in the data. By analysing data changes, organizations can better understand how to improve their processes and operations.
4. **Decision-making:** Data changes can have a significant impact on decision-making. By capturing and analysing data changes, organizations can make more informed decisions based on current and accurate data.

5. **Data governance:** Data governance is the process of managing the availability, usability, integrity, and security of data. Capturing data changes is an important part of data governance, as it helps ensure that data is properly managed, controlled, and secured.

Overall, capturing data changes is critical for maintaining data quality, compliance, performance optimization, decision-making, and data governance. By capturing and analysing data changes, organizations can make more informed decisions, improve their operations, and maintain data integrity.

SCD:

Slowly changing dimensions (SCD) are used to capture data change in data warehousing because they provide a method for tracking changes to dimension data over time. Dimension data is typically used to provide context to fact data, such as sales or revenue data, by describing the attributes of the entities being measured, such as products, customers, or time.

As data changes over time, it is important to track these changes to ensure that the data being used for reporting and analysis is accurate and up-to-date. SCD provides a way to manage changes to dimension data over time, preserving the history of changes and enabling tracking of changes over time. This is essential for data quality and data governance, which are critical for effective decision-making and operations.

SCD is especially useful in situations where there are frequent changes to dimension data, such as customer information, product information, or other attributes that change over time. By using SCD, organizations can capture and track these changes, ensuring that they have access to accurate and up-to-date data.

Slowly changing dimensions (SCD) is a technique used in data warehousing to manage changes to dimensional data over time. In dimensional data, each column represents an attribute and each row represents an instance of a fact. Slowly changing dimensions describe how dimensions change slowly over time and are typically categorized into three types:

1. **Type 0 SCD**: This type of Slowly Changing Dimension (SCD) is a design approach for data warehousing that does not track any changes to dimension data over time. In this approach, the data in the dimension table is overwritten whenever a change occurs, and no history of changes is kept. As a result, Type 0 SCD is also known as "No Change" SCD.
2. **Type 1 SCD**: This type of slowly changing dimension involves overwriting the old data with new data. Type 1 SCD is useful when the old data is no longer needed, and only the latest data is required.
3. **Type 2 SCD:** This type of slowly changing dimension involves creating a new record for each change in the dimension data. This approach preserves the history of the changes and enables tracking of the changes over time. Type 2 SCD is typically used when it is necessary to track changes over time, such as for customer or product information.
4. **Type 3 SCD:** This type of slowly changing dimension involves adding a new column to the table to track changes. Type 3 SCD is useful when only a limited amount of historical information is required and only certain attributes need to be tracked.

TYPE	TITLE	ALLOWS UPDATES	PRESERVES HISTORY	DESCRIPTION
0	Retain Original	✗	✗	Attributes never change (e.g. date of birth).
1	Overwrite	✓	✗	Old values are overwritten with new values.
2	Add a new record	✓	✓	Track changes by creating multiple records (e.g. ValidFrom, ValidTo).
3	Add a new field	✓	✓(*)	Track changes using separate columns (e.g. CurrentValue, PreviousValue).

To implement SCD, data warehousing tools use a combination of ETL (extract, transform, and load) processes, database schema design, and query optimization techniques. ETL processes are used to extract data from source systems, transform it into the required format, and load it into the data warehouse. Database schema design involves designing the database schema to accommodate SCD, such as creating new columns or tables for tracking changes. Query optimization techniques are used to ensure that queries perform efficiently against SCD tables, which can be complex due to the need to join multiple tables.

Overall, slowly changing dimensions are an important concept in data warehousing, as they allow organizations to manage changes to dimensional data over time, which is critical for decision-making, reporting, and analysis. By understanding the different types of SCD and implementing the appropriate approach, organizations can ensure that they have access to accurate and up-to-date data, which is essential for effective operations and decision-making.

Data Description

Telco customer data is a type of data that is collected and stored by telecommunications companies about their customers. This data includes various types of information, such as personal information, usage information, demographic information, device information, and customer service information.

Personal information refers to data that identifies an individual, such as name, address, phone number, email address, date of birth, and social security number. This information is typically collected when a customer signs up for telco services or when they update their account information.

Usage information includes data about a customer's usage of telco services, such as call and messaging logs, internet usage, location data, and billing information. This information is used by telcos to monitor network performance, improve service quality, and offer personalized promotions and recommendations to customers.

Demographic information refers to data about a customer's characteristics, such as gender, age, income, and education level. This data is often used by telcos for market research and to create customer profiles that can be used to target specific customer segments.

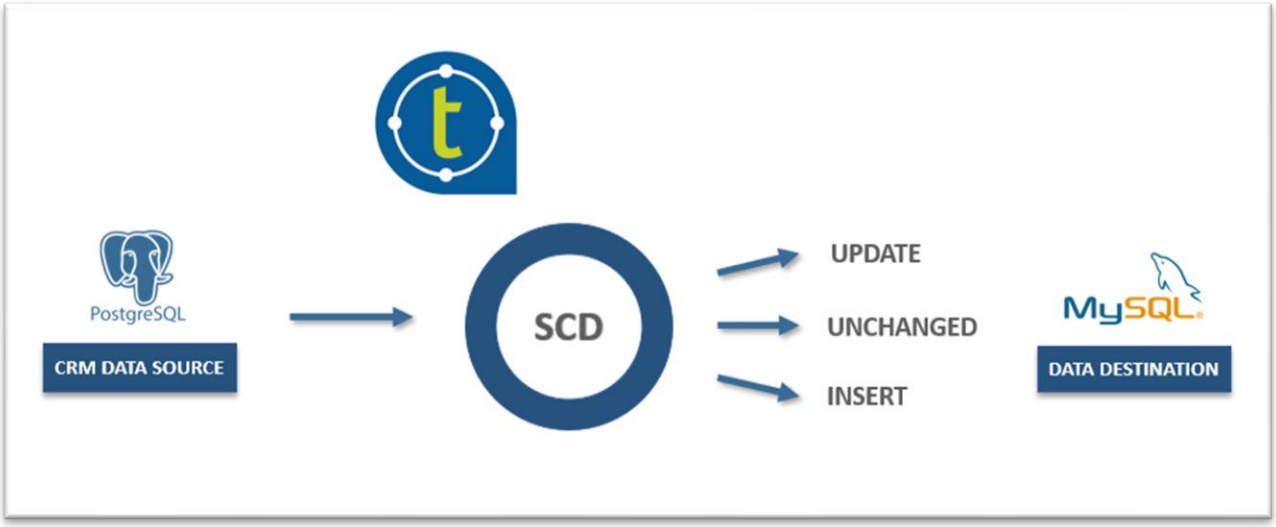
Device information includes data about the devices that customers use to access telco services, such as the type of device, its operating system, and the applications installed on it. This data is used by telcos to optimize their services for different devices and to ensure that customers have a seamless user experience.

Customer service information includes records of customer service interactions, complaints, and inquiries. This information is used by telcos to improve their customer service processes and to address customer concerns.

Telco customer data is typically collected and stored in databases that are managed by the telco. It is important for telcos to handle this data responsibly and to ensure that customer privacy is protected. Telcos must comply with various data protection laws and regulations, such as the General Data Protection Regulation (GDPR) in the European Union and the California Consumer Privacy Act (CCPA) in the United States. Telcos must also take measures to secure customer data and prevent unauthorized access or data breaches.

Use-case depicted in this project

In this project, we will create an ETL pipeline in Talend Open Studio to capture data changes using the Slowly Changing Dimension technique. We will also schedule the task using Task Scheduler to trigger the event after a certain period. This project will help you learn all the basics of SCD, its features, and different types of SCD, along with their implementation with proper use cases.
For example- The employee dimension may hold attributes such as name, date of joining, hour rate, monthly salary gross, and phone number. The employee's details may change over time (e.g.changing hour rate, changing monthly salary gross). A slowly changing dimension can accommodate these changes.



Download the Talend job from here.

Loading data to PostgreSQL database

To load a CSV file to a PostgreSQL database table using Talend, you can follow the steps below:

1. Create a new Talend job and drag the following components to the workspace:
    - tFileInputDelimited: This component reads the CSV file.
    - tPostgresqlOutput: This component writes the data to the PostgreSQL database table.

2. Double-click the tFileInputDelimited component to configure it:
    - In the Basic settings tab, set the File name field to the path of the CSV file.
    - Set the Field separator and Row separator fields to match the format of the CSV file.
    - In the Columns tab, set the schema of the input file. Make sure that the column names and data types match the columns of the PostgreSQL database table.

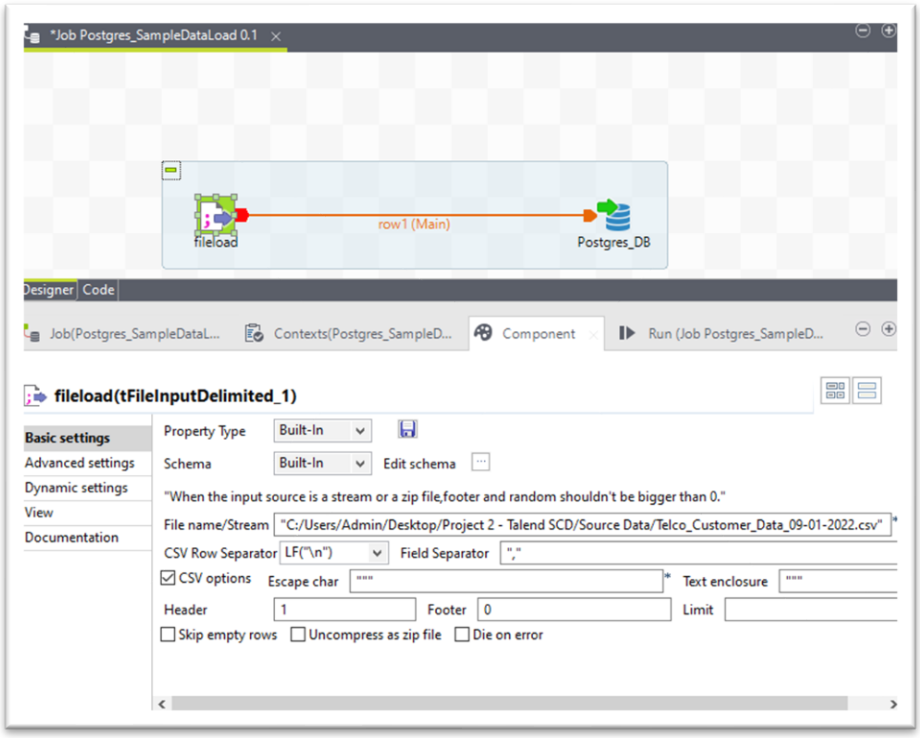
3. Double-click the tPostgresqlOutput component to configure it:
    - In the Basic settings tab, set the Host, Port, Database, Username, and Password fields to connect to the PostgreSQL database.
    - Set the Table field to the name of the target database table.
    - In the Action on table field, select "Insert" to insert new data into the table.
    - In the Mapping tab, set the schema of the target database table. Make sure that the column names and data types match the columns of the input file.

4. Connect the tFileInputDelimited component to the tPostgresqlOutput component by dragging a line between them.

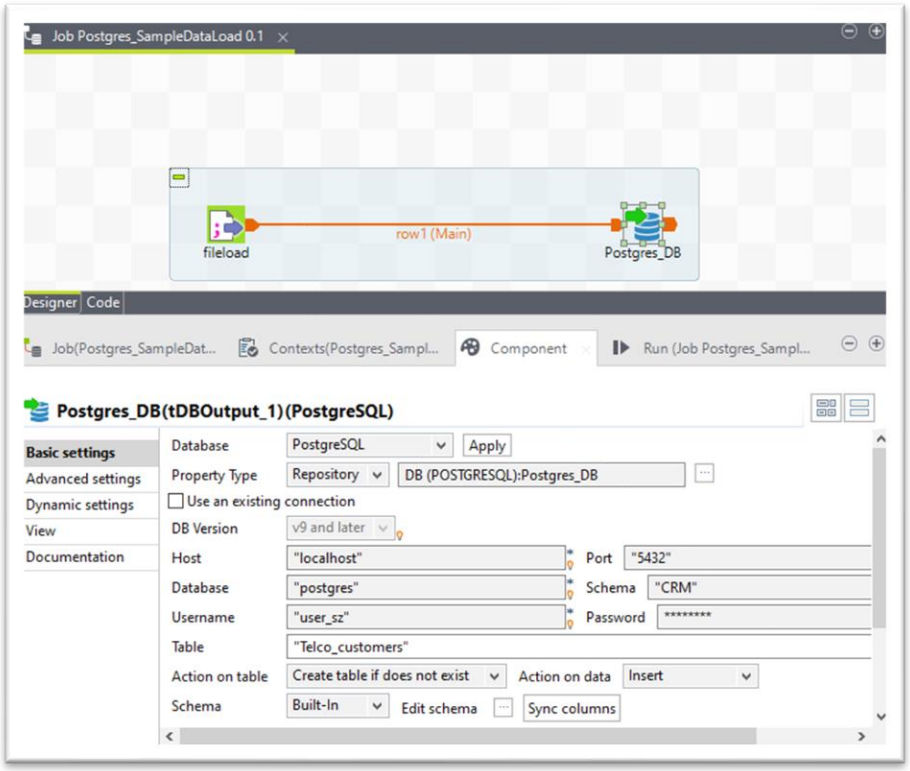
5. Save and run the Talend job.

This should load the data from the CSV file to the PostgreSQL database table.

Configurations of tFileInputDelimited component:

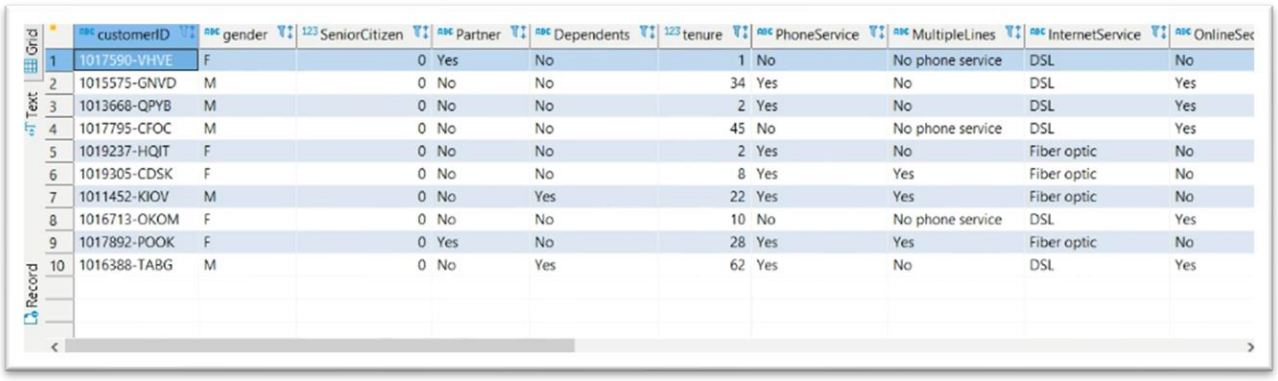


Configurations of tPostgresqlOutput component:



Run this Talend job.
Go to PostgreSQL, and execute this query:

Output:



	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSec
1	1017590-VHVE	F	0	Yes	No	1	No	No phone service	DSL	No
2	1015575-GNVD	M	0	No	No	34	Yes	No	DSL	Yes
3	1013668-QPYB	M	0	No	No	2	Yes	No	DSL	Yes
4	1017795-CFOC	M	0	No	No	45	No	No phone service	DSL	Yes
5	1019237-HQIT	F	0	No	No	2	Yes	No	Fiber optic	No
6	1019305-CDSK	F	0	No	No	8	Yes	Yes	Fiber optic	No
7	1011452-KIOV	M	0	No	Yes	22	Yes	Yes	Fiber optic	No
8	1016713-OKOM	F	0	No	No	10	No	No phone service	DSL	Yes
9	1017892-POOK	F	0	Yes	No	28	Yes	Yes	Fiber optic	No
10	1016388-TABG	M	0	No	Yes	62	Yes	No	DSL	Yes

We have successfully loaded data into PostgreSQL database using a Talend job.

Execution of SCD Type 2

Slowly Changing Dimensions (SCD) are a common concept in data warehousing, which refers to the way dimensional data changes over time. Dimensional data in a data warehouse is often described by several attributes or columns, and these attributes can change over time. SCD type 2 is one of the most commonly used SCD types, and it tracks both historical and current data.

The SCD type 2 strategy tracks historical changes to dimension data by creating a new row for every change that occurs in any of the attributes. It creates a new row in the dimension table with a new primary key, which represents the new version of the entity with updated attributes, and retains the old row to represent the previous version of the entity with a valid-from and valid-to time interval.

For example, let's consider a customer dimension table with the following columns: customer_id, name, address, and start_date. If the address of a customer changes from "123 Main St" to "456 Oak Ave" on January 1, 2022, the SCD type 2 strategy will create a new row with a new primary key, a new start date of January 1, 2022, and the updated address value of "456 Oak Ave". The original row for the customer will be updated with the end date of December 31, 2021, which corresponds to the start date of the new row. This way, the dimension table accurately reflects the historical data, and the current data can be easily obtained by querying the table for the rows with a valid-to date of "9999-12-31".

Implementing SCD type 2 in a data warehouse helps maintain data integrity and provides a complete historical record of the changes that occur over time, making it easier to analyze and understand trends and patterns in the data. It allows the data warehouse to keep track of changes to the dimension data and provides a complete historical record of the changes that occurred over time, which is essential for accurate and efficient reporting and analysis.

To implement an SCD type 2 strategy, the following columns are typically added to the dimension table:

- A primary key column that uniquely identifies each row in the table.
- Valid from and valid to columns that represent the start and end dates for the row, respectively.
- Current indicator column that identifies the current row in the dimension.

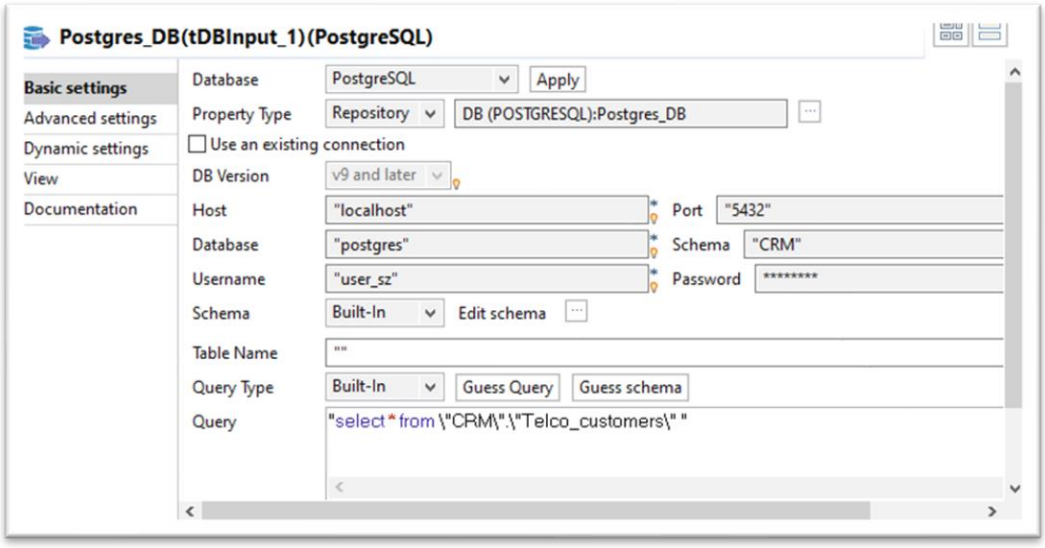
In Talend, SCD type 2 can be implemented using the tDBSCD component. This component supports various SCD types and can be configured to create and update dimension tables in a data warehouse.

SCD Type 2 Talend Job:

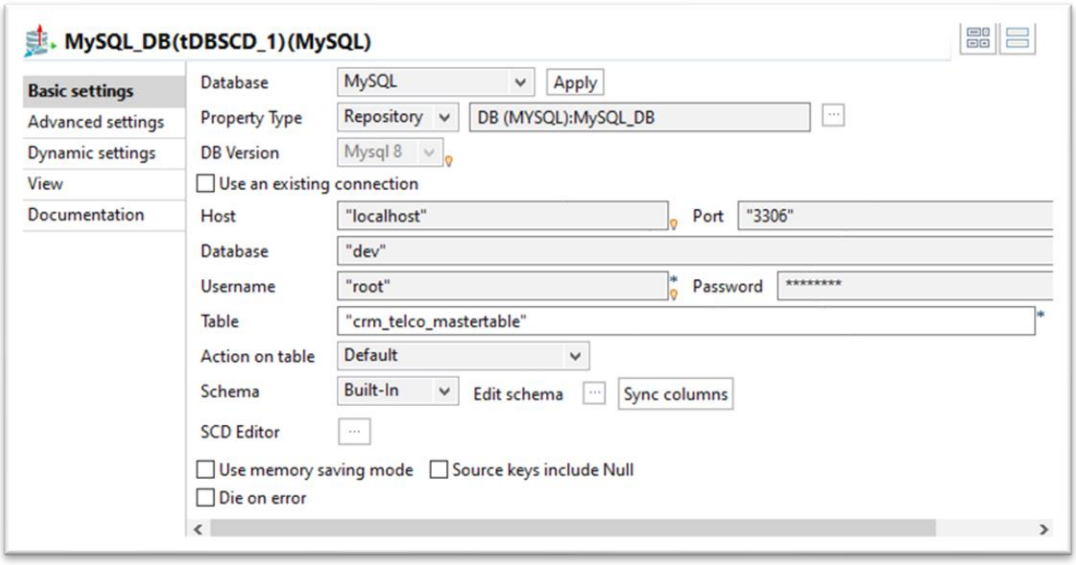
To create a Talend job to implement a Slowly Changing Dimension (SCD) type 2 from PostgreSQL to MySQL using the tDBSCD component, you can follow the steps below:

1. Create a new Talend job and drag the following components to the workspace:
    - tPostgresqlInput: This component reads the data from the PostgreSQL database table.
    - tDBSCD: This component applies the SCD transformation and writes the data to the MySQL database table.

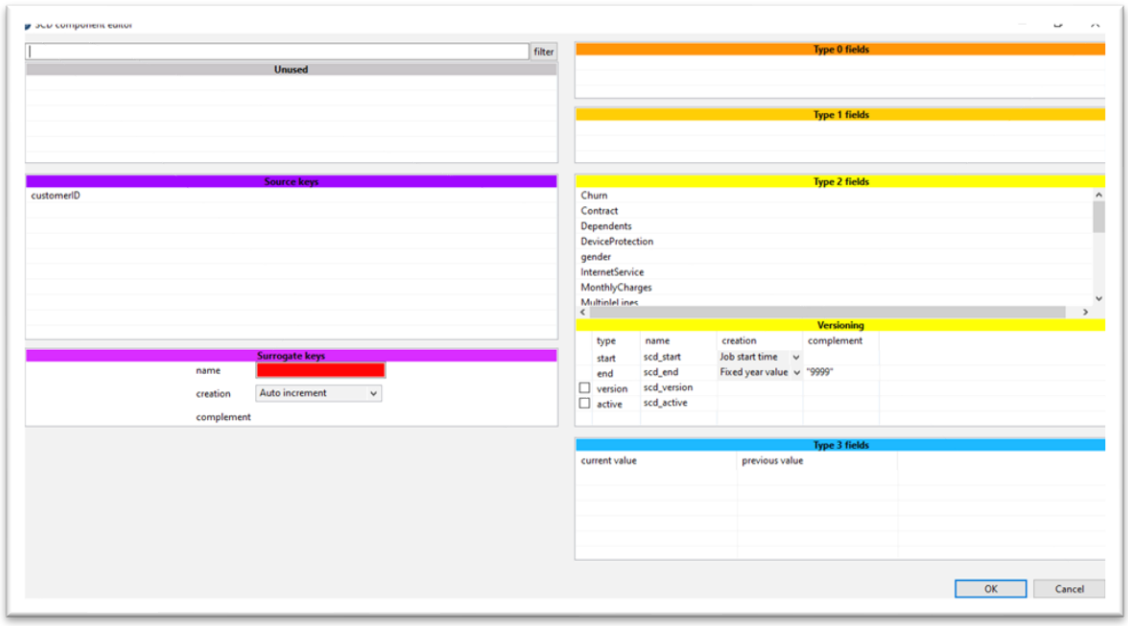
2. Double-click the tPostgresqlInput component to configure it:
    - In the Basic settings tab, set the Host, Port, Database, Username, and Password fields to connect to the PostgreSQL database.
    - Set the Table Name field to the name of the source database table.
    - In the Query field, write a SQL query to select the data from the source table.



3. Double-click the tDBSCD component to configure it:
    - In the Basic settings tab, set the Host, Port, Database, Username, and Password fields to connect to the MySQL database.
    - Set the Table Name field to the name of the target database table.
    - Click on SCD Editor



- Set the Key Field Name field to the name of the primary key column in the target table.



Here, customerID is the primary key, hence we drag it in Source keys, rest of all columns put it in Type 2 fields.

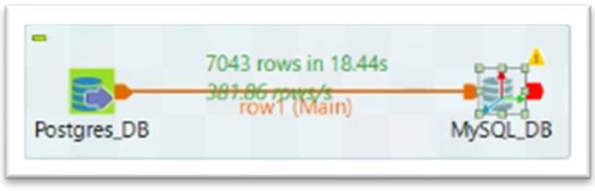
- In Versioning, we select Job start time for scd_start value and a Fixed value for scd_end value.
- Click ok.

4. Before we run this job, we need to create table schema in MySQL to store the data coming from PostgreSQL.

Script:

```
CREATE TABLE dev.`crm_telco_mastertable` (
  `customerID` varchar(12) ,
  `gender` varchar(1) DEFAULT NULL,
  `SeniorCitizen` varchar(10) DEFAULT NULL,
  `Partner` varchar(3) DEFAULT NULL,
  `Dependents` varchar(3) DEFAULT NULL,
  `tenure` varchar(10) DEFAULT NULL,
  `PhoneService` varchar(3) DEFAULT NULL,
  `MultipleLines` varchar(16) DEFAULT NULL,
  `InternetService` varchar(11) DEFAULT NULL,
  `OnlineSecurity` varchar(19) DEFAULT NULL,
  `OnlineBackup` varchar(19) DEFAULT NULL,
  `DeviceProtection` varchar(19) DEFAULT NULL,
  `TechSupport` varchar(19) DEFAULT NULL,
  `StreamingTV` varchar(19) DEFAULT NULL,
  `StreamingMovies` varchar(19) DEFAULT NULL,
  `Contract` varchar(14) DEFAULT NULL,
  `PaperlessBilling` varchar(3) DEFAULT NULL,
  `PaymentMethod` varchar(25) DEFAULT NULL,
  `MonthlyCharges` varchar(8) DEFAULT NULL,
  `TotalCharges` varchar(8) DEFAULT NULL,
  `Churn` varchar(3) DEFAULT NULL,
  scd_start datetime,
  scd_end date,
  primary key(customerID,scd_start),
  index ind (scd_start,scd_end)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

5. Connect the tPostgresqlInput component to the tDBSCD component by dragging a line between them.
6. Save and run the Talend job.



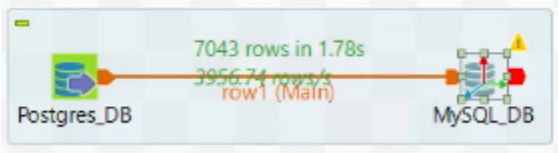
This should implement the SCD type 2 strategy from PostgreSQL to MySQL using the tDBSCD component in Talend. The component will create a new row for each change in the input data and retain the old row with a valid-to time interval.

Testing the Job:

1. Update a few records in PostgreSQL.

```
Update "CRM"."Telco_customers" set "InternetService" = 'Fiber optic' where "customerID" in ('1017590-VHVE', '1017795-CFOC')
```

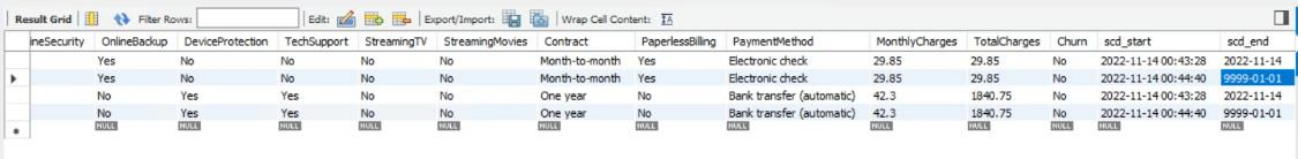
2. Run the Talend Job.



3. Run the following query in MySQL.

```
Select * from dev.crm_telco_mastertable where customerID in ('1017590-VHVE', '1017795-CFOC')
```

Query result:



ineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn	scd_start	scd_end
	Yes	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No	2022-11-14 00:43:28	2022-11-14
	Yes	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No	2022-11-14 00:44:40	9999-01-01
	No	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.3	1840.75	No	2022-11-14 00:43:28	2022-11-14
	No	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.3	1840.75	No	2022-11-14 00:44:40	9999-01-01

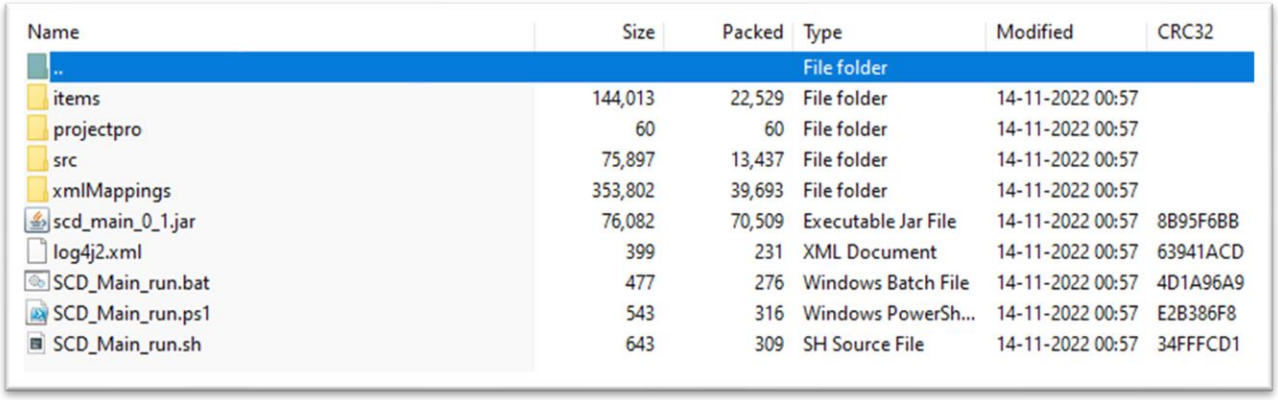
We can see, new rows are created for each change in the input data and retain the old row with a valid-to time interval.

Build a Talend Job

To build a job in Talend Open Studio, follow these steps:

1. Open your Talend Open Studio project and select the job you want to build.
2. Right-click on the job and select "Build Job" from the context menu.
3. In the Build Job wizard, select the destination folder where you want to save the job.
4. Click "Finish" to build and export the job.

The exported job will be saved in the specified destination folder in your selected format. It will create a zip folder. Once you unzip, you will see the following files:



Name	Size	Packed	Type	Modified	CRC32
..			File folder		
items	144,013	22,529	File folder	14-11-2022 00:57	
projectpro	60	60	File folder	14-11-2022 00:57	
src	75,897	13,437	File folder	14-11-2022 00:57	
xmlMappings	353,802	39,693	File folder	14-11-2022 00:57	
scd_main_0_1.jar	76,082	70,509	Executable Jar File	14-11-2022 00:57	8B95F6BB
log4j2.xml	399	231	XML Document	14-11-2022 00:57	63941ACD
SCD_Main_run.bat	477	276	Windows Batch File	14-11-2022 00:57	4D1A96A9
SCD_Main_run.ps1	543	316	Windows PowerSh...	14-11-2022 00:57	E2B386F8
SCD_Main_run.sh	643	309	SH Source File	14-11-2022 00:57	34FFFCD1

We will use the Windows Batch File to schedule this job on the Task scheduler.

Schedule the job using Windows Task Scheduler

To schedule a Windows batch file using Windows Task Scheduler, follow these steps:

1. Open the Windows Task Scheduler by pressing the Windows key + R, typing "taskschd.msc" and hit Enter.
2. Click "Create Basic Task" from the Actions panel on the right side.
3. In the Create Basic Task wizard, provide a name and description for the task.
4. Select the trigger for the task. You can run the task daily, weekly, monthly, or when the computer starts.
5. Select the start time for the task and any other relevant options, such as the end time or the interval between repeats.
6. In the Action section, select "Start a program" and click "Next".
7. In the Program/script field, enter the path to the batch file, e.g., 'C:\Project\SCD_Main_run.bat'.
8. Click "Next" and "Finish" to complete the task setup.
9. The task will run at the specified schedule and execute the batch file.

ProjectPro

Summary

- A Talend Real-Time Project for ETL (Extract, Transform, Load) Process Automation is a project designed to automate the process of extracting data from various sources, transforming it into a desired format, and loading it into a target system, such as a data warehouse. The Talend Open Studio is used to develop and execute the ETL job, which can handle large volumes of data and process it in real-time.
- The prerequisites for this project include a basic understanding of data warehousing concepts, experience with SQL, and knowledge of Talend Open Studio components.
- Understood the need to capture data change.
- Understood the slowly changing dimensions phenomenon and its various types.
- We built a Talend job to capture data change using SCD type 2.
- The job can be exported as a standalone Java application or a command-line executable, and scheduled to run automatically using Windows Scheduler.