Supply Chain Twin Documentation

GOOGLE CONFIDENTIAL

Table of Contents

Introduction

The Supply Chain Twin purpose-built solution enables supply chain professionals to build a digital representation of their physical supply chain by organizing and orchestrating data from across the entire supply chain. This will provide end-to-end visibility across the supply chain and help optimize planning and decision making across supply chain functions.

The solution is single-tenant i.e., deployed within the customer's Cloud environment. Deployments typically include data from the customer's business and operational systems and business partners. Additionally, it includes publicly available contextual data such as weather, relevant to model the physical supply chain.

The solution serves as the data foundation upon which other Supply Chain solutions such as Pulse and Simulations are developed. Finally, we work with our partners to develop their offerings using Supply Chain Twin as the foundation.

Solution Overview

Modeling the physical supply chain requires data from a multitude of sources. These can vary substantially by use cases, however, can be classified into three segments based on characteristics and source:

1. **Private Segment**: Comprising data from the customer's business and operational systems and applications.
2. **Community Segment**: Comprising data from a trusted community of suppliers, customers, and partners.
3. **Public Segment**: Consisting of contextual data such as weather, sustainability, risk, and so on.

The digital representation of the physical supply chain is therefore enabled as a composable model that cuts across these segments.

The Supply Chain Twin users have the capability to select the different BigQuery datasets to create their unique representation of their supply chain. Each of the three segments is eventually realized as datasets in BigQuery. BigQuery provides strong governance and access control mechanisms.

This solution implementation guide outlines how to bring together Private Segment, Community Segment, and Public Segment Data from an already published list of Public Data sets.

The Private Segment and Canonical Data Model

The solution provides a Private Segment canonical data model comprising Locations, Products, Legal Entities, Assets, Inventory, Forecasts, and Orders. This representation is domain-oriented, and agnostic of the data sources which are used for the data. For example, the solution provides connectors and pipelines needed to bring in the data from the SAP ERP (S4/HANA), however, these pipelines can be adapted or additional ones built to accommodate other data sources.

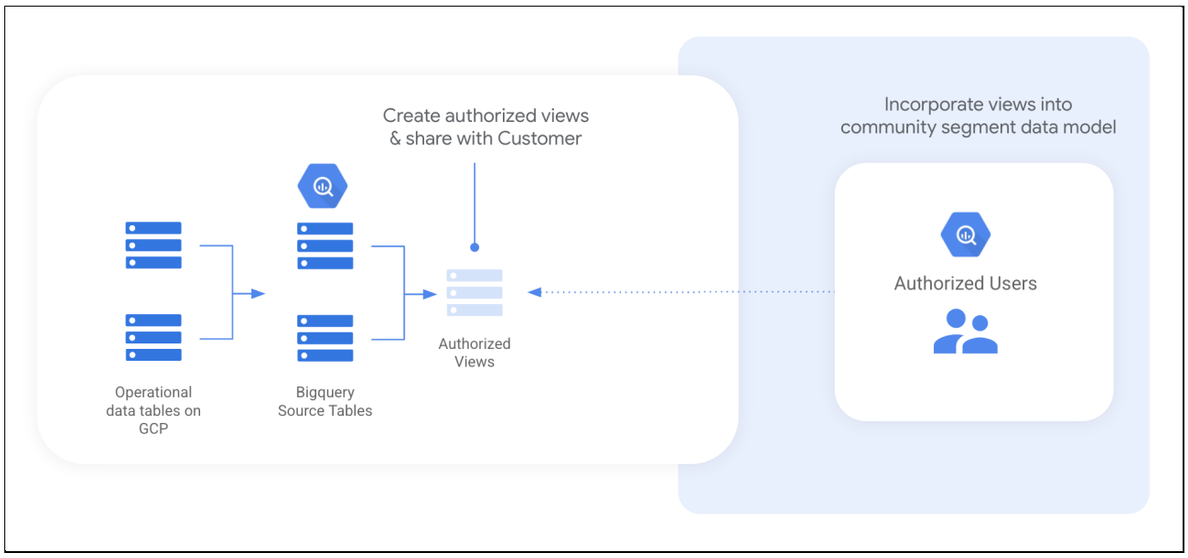
The canonical model is also designed to be industry-agnostic, and lightweight, enabling building KPIs and alerts for visibility. This makes it reusable across multiple use cases. Other solutions for visualization or collaboration utilize this data model as the foundation.

ETL pipelines built using Cloud Data Fusion provide the capability for batch and incremental loading of the staging and dimensions/facts layer of the BigQuery data model. This reduces the tens of tables and hundreds of columns pulled from SAP into the lightweight schema needed for visibility. Similarly, additional schemas needed for specialized AI algorithms (Inventory Optimization, Demand Forecasting, etc) are derived from the dimensions/facts layer. This keeps modules loosely coupled while still relying on a common data foundation as the basis.

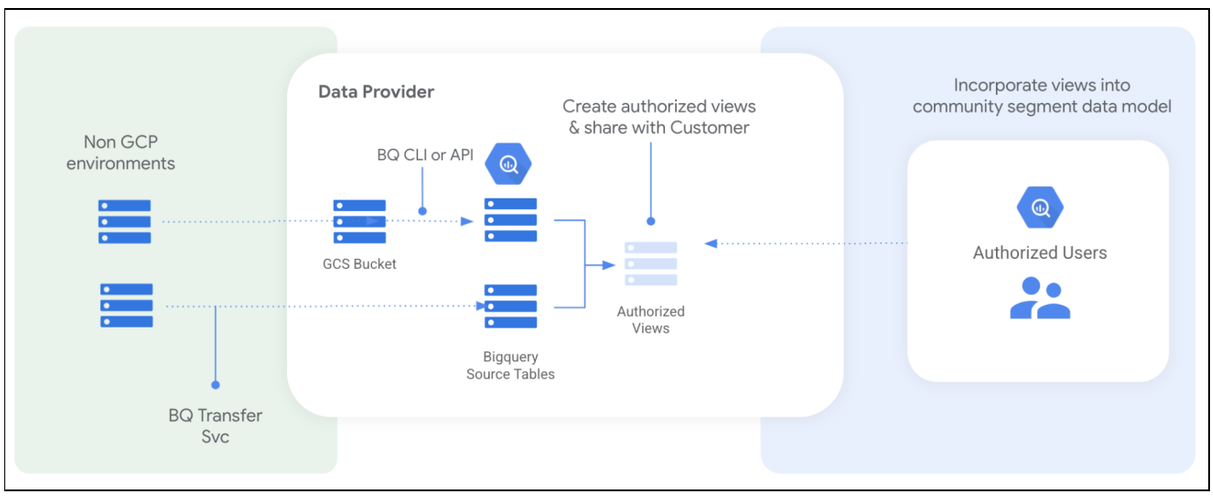
The Community Segment

The community segment is a means to enhance collaboration across a network of partners: suppliers, logistics providers, carriers, freight forwarders etc. The Supply Chain Twin provides the ability to simply share data bidirectionally among partners directly within BigQuery, avoiding the need for complex integrations. Data shared in this manner remains within the control of the data provider, while instantly making it available to data consumers.

BigQuery provides strong security controls to share Datasets and Tables through standard IAM policy enforcement. BigQuery Authorized Views provide the controls needed to share filtered views of data without sharing the underlying tables. Data provider and consumer BigQuery tenancies need to be in the same Google Cloud region.



Multiple options are available when source data is hosted on platforms other than GCP. The BigQuery CLI and API provide means to load the data directly into the consumer's BigQuery tenancy. BigQuery Data Transfer Service provides means to transfer data from AWS S3 or Redshift as well as a multitude of environments.



The Supply Chain Twin also leverages the Community Segment approach to provide integrations into key ISV systems, so the data model shared from these systems becomes composable into the overall data model, and the data from these systems becomes immediately available.

The Public Segment

The Google Cloud Platform hosts hundreds of public datasets. The Supply Chain Twin makes a curated set of public datasets available to provide the contextual data needed to model the physical supply chain.

Public (free) datasets		
Category	**Dataset Name**	**Provider**
Weather	Severe Storm Event Details	NOAA
Weather	Preliminary Storm Reports	NOAA
Weather	Global Forecast System	NOAA
Weather	GSOD	NOAA
Maps	Census Bureau US Boundaries	United States Census Bureau
Maps	US Roads	United States Census Bureau
Healthcare	COVID-19 Public Forecasts	BigQuery Public Datasets Program
Healthcare	COVID-19 Open Data	BigQuery Public Datasets Program
Healthcare	NY Times US Coronavirus Database	The New York Times
Healthcare	COVID-19 Search Trends symptoms	BigQuery Public Datasets Program
Weather & Satellite Imagery		Google Earth Engine

Commercial Datasets		
Category	**Dataset Name**	**Provider**
Risk	Supplier Details and Risk	Craft.co
Sustainability	Sustainability Reference (Factset)	Factset
Sustainability	Supply Chain Reference (Factset)	Factset
Sustainability	Carbon emissions (Morningstar)	Morningstar
Climate and Satellite imagery	Climate Engine	Climate Engine
Social	Public Data Series	OECD, EUROSTAT

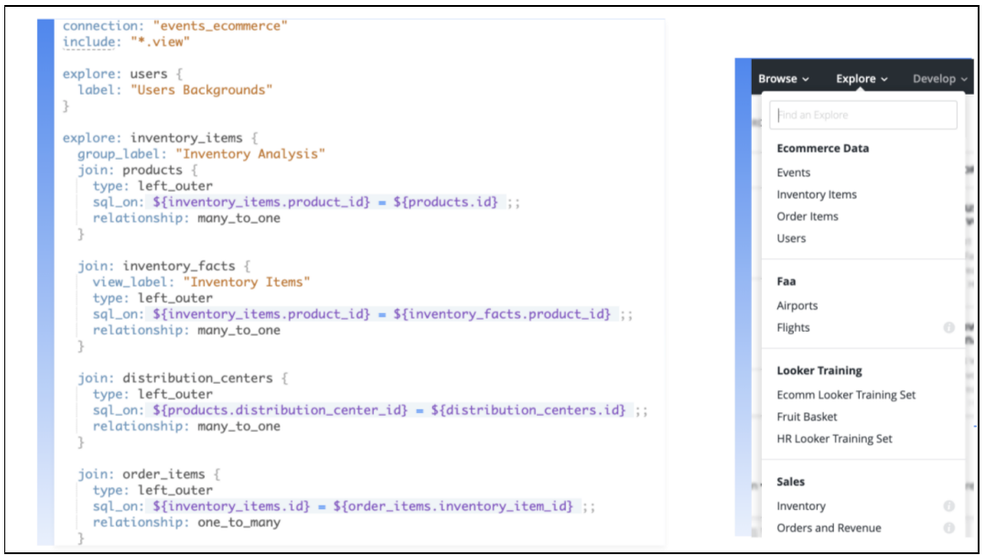
Additional commercial datasets are continuously being evaluated, and will be incorporated into the Public segment.

The Semantic Layer

The overall data model of the Supply Chain Twin is composed using entities from the Private, Community and Public Segments. The Semantic Layer of the Twin, implemented using Looker's LookML modeling language, provides a unified, domain-centric means to utilize this composite data model for analytics. This layer is deployed within and accessed from Looker.

The Semantic layer included entities across all three segments, including views that join entities across the segments allowing for analytics that span the segments. Specific examples of views implementing such joins are available by default and can be extended to other views depending on the datasets that are unique to the environment.

The LookML layer also includes KPI specifications that have been implemented by default.



Solution Deployment

Solution Accelerators

This section of the document outlines the installation process to utilize the solution accelerators to deploy The Supply Chain Twin within an environment. The accelerators include:

- SAP Data Fusion Connectors: ERP integration to BigQuery including ETL pipelines
- Canonical Data model (customizable): Canonical/foundation BigQuery data model that can be customized for customer environments
- LookML Semantic Model covering core data model entities (customizable): For use with Looker capabilities
- Pre-defined integration with select ISV partners: for community segment data
- Automation: for setup and management

All of these are available as a Google Cloud Source Code repository.

Required skills

- Expertise in SAP on-prem ERP systems and configuration
- Familiarity with the Cloud Data Fusion
- Familiarity with BigQuery
- Familiarity with Looker
- Familiarity with Identity and Access Management (IAM) service accounts and access control
- Familiarity with Terraform
- Familiarity with Cloud Composer
- Familiarity with Source Repositories
- Familiarity with data analytics, including writing SQL queries
- Familiarity with Kimball's dimensional data model

Required users

The configurations described on this page require changes in your SAP system and Google Cloud. You need to work with the following users of those systems to perform the configurations:

User type	Description
SAP admin	Administrator for your SAP system who can access the SAP service site for downloading software.
SAP user	An SAP user who is authorized to connect to an SAP system.
GCP admin	The administrator who controls IAM access for your organization, who creates and deploys service accounts and grants permissions for Cloud Data Fusion, BigQuery, and Looker.

Cloud Data Fusion user	Users who are authorized to design and run data pipelines in Cloud Data Fusion.
BigQuery Data Owner	Users who are authorized to create, view, and modify BigQuery datasets.
Cloud Composer Worker	Provides the permissions necessary to run a Cloud Composer environment VM. Intended for service accounts.
Looker Developer	These users have access to their own Looker Instance. The Looker artifacts are cloned from the source repository and managed in their own git repository. They must develop, manage_model, and deploy permissions.
<Automation User>	Users who are authorized to run the Terraform Automation scripts on the target GCP project. Users will launch Cloud Console and execute scripts cloned from source repository

Required IAM roles

In the solution's sample implementation, you might need additional roles if your project relies on other Google Cloud services. Following IAM roles are required:

- BigQuery Data Owner (roles/bigquery.dataOwner)
- Storage Object Viewer (roles/storage.objectViewer)
- Cloud Data Fusion Runner (roles/datafusion.runner) needs to be granted to the Dataproc service account.
    - Data Fusion Admin: roles/datafusion.admin
- Cloud Composer needs following permissions to be granted for orchestration and scheduling of the data pipelines.
    - Project Editor: `roles/editor`
    - Network Admin: `roles/compute.networkAdmin`
    - Instance Admin: `roles/compute.instanceAdmin.v1`
    - Service Account User: `roles/iam.serviceAccountUser`
    - Composer Worker: `roles/composer.worker`
- Access to Supply Chain Twin Project project git repository (permissions) to access delivered artifacts including CDF pipeline definitions, LookML semantic layer, and Terraform scripts.

Automation

Supply Chain Twin solution deployment is a combination of automated and manual steps. This section details how you can set up the Supply Chain Twin (using SAP as a source system) to the Private Data Set.

Terraform Automation scripts have been developed for the following key areas:

- GCP Project configuration for services: The key services that need to be made available for the solution to be ready for use are:
    - Cloud Data Fusion: An instance of Cloud Data Fusion is set up in the GCP environment.
    - Setting up the BigQuery datasets and schemas.
    - Setting up a Cloud Composer Environment.
- Data integration
    - Deploying pre-designed pipelines to Cloud Data Fusion.
    - Orchestrating the execution of the pipelines for data load.
    - Scheduling runs of these pipelines for operational use.

Before you begin

1. Create a GCP Project which will host the Twin
2. Launch Cloud Shell within the Project to continue with the rest of the installation
3. Clone Git Repo using Google Cloud SDK option
4. Set up Cloud Composer with apache-airflow-providers-google v5.1.0 or above.

5. Ensure network connectivity to the SAP source system from the GCP Project.
6. Creating the Copy of the repo to github

Stage the automation scripts

1. Clone the Git Repository
    a. gcloud init
    b. gcloud source repos clone scl-dev --project=dev-cs-1
    c. cd `scl-dev/automation`
2. Ensure the following directories exists
    a. 01_Before_Installation
    b. 02_Create_CDF_Instance
    c. 03_Dataproc_Profile
    d. 04_BigQuery_Dataset
    e. 05_Composer_Environment
    f. 06_Deploy
    g. 07_Orchestration
3. In each directory, consult the readme to familiarize yourself with the process and customize the scripts to your environment's needs, including setting the variables
4. Creating the Copy of the repo to github
    a. Before you start
        i. Familiarize with cloning GCP Source using gCloud
        ii. **source2git** is the name of the subfolder where the private git repository will be set up
        iii. **reponame** is the name of the private repository to be created. This repository will be used to store the artifacts that are cloned from the **scl-dev** source repo.
    b. Below are the commands to create a copy of the published artifacts from the scl-dev repo
        i. $ mkdir source2git
        ii. $ cd source2git
        iii. $ gcloud source repos clone scl-dev --project=dev-cs-1
        iv. $ git clone https://github.com/username/reponame
        v. $ cd reponame
        vi. $ cp -R ../scl-dev/* .
        vii. $ git add .
        viii. $ git commit -m "your remark"
        ix. $ git push origin branchname

Run book

Overview

You can implement the Supply Chain Twin in your project with the following steps:

1. Configure the SAP ERP system and install the SAP transport provided.
2. Set up your Cloud Data Fusion environment to use the SAP ODP Batch Source plugin.
3. Create datasets in BigQuery. The solution provides datasets for staging, dimensional, and fact tables.
4. Clone the git repository to the Cloud Shell to get the pre-built CDF pipelines that bring data from the available SAP Source.
5. Deploy CDF pipelines, and configure SAP connection parameters and SAP ODP Batch Source plugin..
6. Run Cloud Composer to load at scheduled times for full initial data load full and progressive delta data loads to the target BigQuery datasets.
7. Connect Looker to the BigQuery dataset.
8. Deploy the Looker LookML file in Looker instance for the semantic layer definition of Supply Chain Twin.

For more information on using CDF for data integration, see Using the SAP ODP Batch Source plugin.

Steps

Note: The automation scripts have been tested with CDF v6.4.0 and ODP plugin version 0.0.12. This procedure will be progressively improved as later versions of services and improvements in automation are released.

Step	Description	Sample Commands / Remarks
01_Before_installation: Set up the environment	Terraform, gCloud SDK have to be ready in the local machine	
[Optional] **Clean terraform state files**:	For repeated runs these files have to be deleted	`rm -rf terraform.tfstate` `rm -rf .terrafor*`

02_Create_CDF_Instance	In this step, the latest version of CDF is installed. Adapt the terraform scripts to choose a specific version	`terraform init`
		`terraform plan`
		`terraform apply`

[Optional]**03_Dataproc_Profile**	Creating a new profile for the existing Dataproc cluster and associating it with CDF Instance	`terraform init`
		`terraform plan`
		`terraform apply`

[Optional]**Manual:** Upload Plugin	If the latest plugin is available, upload it to the CDF instance	This is useful when a bugfix version that is not yet available on the CDF Hub needs to be used.
Manual: VPC Peering with SAP Network	CDF Network has to peer with SAP Instance	In case SAP and CDF are in the same GCP project this is a one-time peering. In case SAP and CDF are across two GCP projects, then peering needs to be set up on both GCP projects.
Manual: Set up SAP Secure Macros	Set up once at CDF instance level to be used by all pipelines	
04_BigQuery_Dataset	Sets up the destination datasets in BigQuery for pipelines	The target datasets can be customized in the variables.tf file.
05_Composer_Environment	Sets up the Composer Environment to set up the orchestration scripts.	Ensure that the PyPi package has the `apache-airflow-providers-google / >=5.1.0 set.`
06_Deploy	Deploys the pipeline to the CDF instance based on the path to the pipelines + l0-staging-full + l1-staging-delta	`+ l0-staging-full deploy_fullload.sh` `+ l1-staging-delta deploy_delta.sh`
07_Orchestration **Manual:** Upload orchestration scripts	Use the Cloud Composer environment to be used and upload the orchestration scripts	
Manual: Trigger the execution	Validate the pipelines are running in the CDF Instance	

Semantic layer

Looker defines the semantic layer for the solution by providing a pre-configured LookML model. This file corresponds to the Canonical Layer of Supply Chain Twin solution. For the Supply Chain Twin solution, the corresponding artifacts are maintained in the git repo.

1. Go to the directory where looker artifacts are present:
   a. `cd scl-dev : manifest.lkml`
   b. `cd scl-dev/looker : all files and folders`
2. Copy these artifacts to the corresponding Looker Git repo.
3. Modify the data source name appropriately to point to the datasets in BigQuery.
   a. To connect Looker to BigQuery, see the Looker documentation about BigQuery connections.

What's next

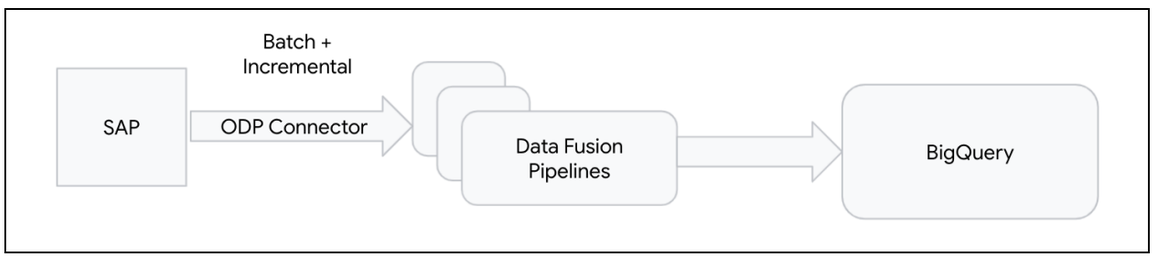
- Learn more about Cloud Data Fusion.
- Learn more about SAP on Google Cloud.
- Learn more about BigQuery.
- Learn more about Looker Blocks.
- Learn more about Cloud Composer

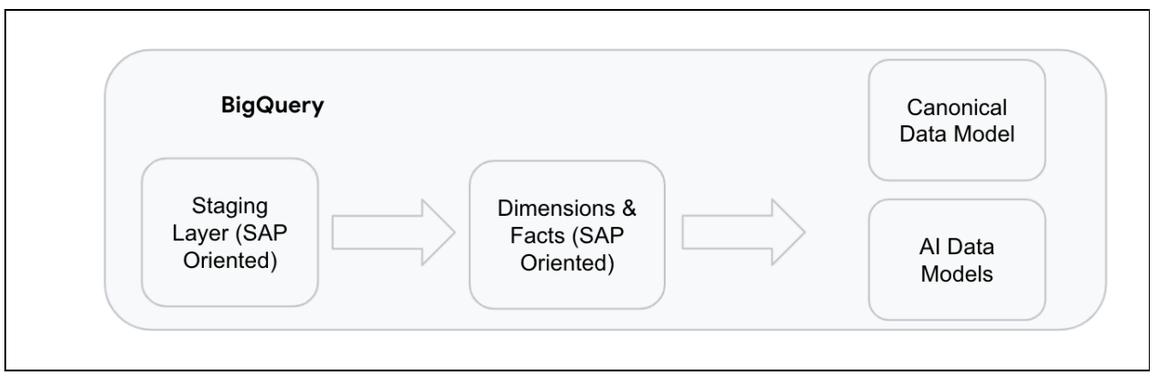
Appendix

SAP integration

There are three main components that bring the data from SAP to make it accessible to the Supply Chain Twin Solution:

- SAP Source System ODP extractors
- Cloud Data Fusion SAP ODP Connector
- Cloud Data Fusion Pipelines
    - Staged data (L0) layer that represents raw SAP data
    - Transformed facts and dimension tables (L1)
    - Supply Chain Twin canonical data model(L2)





Private Segment datasets in BigQuery

The following datasets are created in BigQuery:

Description	Examples
Contains all the tables from the SAP Source system as identified for that business process.	`scl_sap_staging`
Contains the key dimension entities like Customer Dimension, Supplier Dimension, Plant Dimension, Material Dimension.	`scl_sap_dimension`
Contains the fact tables generated from the pipeline.	`scl_sap_fact`
Maps to the Business Entities as identified by the Twin Team.	`scl_canonical`

These names are customizable as noted in the run book section.

Pipelines in Cloud Data Fusion

There are four types of CDF pipelines:

- **Staging layer pipelines(L0}**: The staging dataset in this type of pipeline is a direct mapping to the original source table in SAP. The sample staging layer pipelines have names that refer to the SAP source table and the BigQuery target table. For example, a pipeline named KNA1_Customer_Master refers to the SAP Source Table (`KNA1`) and BigQuery target table (`CustomerMaster`). These tables are created in the BigQuery dataset `scl_sap_staging`.
- **Dimension layer pipelines(L1)**: The dimension layer dataset in this type of pipeline is a curated and refined version of the staging dataset that creates the dimension and facts needed for the analysis. The sample pipelines have names that refer to the target entity in the target BigQuery dataset. For example, a pipeline called customer_dimension refers to the Customer Dimension entity in the BigQuery dataset `scl_sap_dimension`.
- **Fact layer pipelines(L1)**: The fact layer dataset is a curated and refined version of the staging dataset that creates the facts that are necessary for the analysis. These sample pipelines have names that refer to the target entity in the target BigQuery dataset. For example, a pipeline called sales_order_fact delivers curated data to the Sales Order Fact entity in the corresponding BigQuery dataset `scl_sap_fact`.
- **Canonical layer pipelines(L2)**: The facts and dimensions contribute to the data in the canonical layer that enables the twin solution to work with other data sets such as community data sets or public data sets.

The following sections summarize how to get the pipelines to work in your environment:

Configure staging layer pipelines

There are two configuration steps for the staging pipelines:

1. Configure the source SAP system.
2. Configure the target BigQuery dataset and table.

Parameters for the SAP ODP plugin

The SAP Table Batch Source plugin reads the content of an SAP table or view. The solution provides the macros, which you can modify to control your SAP connections centrally. More information on the macros and their description can be found here.

Macro name	Description	Example
`${SAPClient}`	SAP client to use	100
`${SAPLanguage}`	SAP logon language	EN
`${SAPApplicationServerHost}`	SAP server name or IP address	10.132.0.114
`${SAPSystemNumber}`	SAP system number	00
`${secure(saplogonusername)}`	SAP User name	For more information, see Using Secure Keys.
`${secure(saplogonpassword)}`	SAP user password	For more information, see Using Secure Keys.

For more information, see Configuring the plugin.

Parameters for the BigQuery Pipelines

The solution provides the following macros for BigQuery targets are set appropriately on the different pipelines

Macro name	Description	Example

`${Dataset}`	Target dataset	`scl_sap_staging`
`${ProjectID}`	The project ID where the BigQuery dataset has been created.	`sap-adapter`
`${StagingDatasetName}`	Source dataset from staging layer	`scl_sap_staging`
`$(TargetDatasetNames1)`	Target dataset for dimension pipelines	`scl_sap_dimension`
`$(TargetDatasetNames2)`	Target dataset for fact pipelines	`scl_sap_fact`
`$(TargetDatasetNames3)`	Target dataset for canonical pipelines	`scl_canonical`

Deploying Cloud Data Fusion pipelines

Data Integration pipelines have been developed for bringing Data from SAP using ODP extractors using CDF. These pipelines have been developed based on data warehousing principles for full and incremental loads to bring data into the data staging layer (L0). The next layer related to facts and dimensions is generated using (L1) pipelines. The final set of pipelines transforms the dimensional data model to Supply Chain Twin canonical model (L2). All the target data sets are created in BigQuery.

Details of SAP ODP Sources and CDF pipelines are listed in the Supply Chain Twin Git Repository. The details of the data model are available here.

The final scl_canonical dataset is now ready to be integrated into the Supply Chain Twin semantic layer implemented using LookML models.