

PROOF OF CONCEPT

Forecast Shopify Sales with SAP BDC Seamless Planning

Implementation Guide and Lessons Learned

Use SAP BDC Seamless Planning to integrate real time sales data into planning & forecast versions. Avoid data duplication, recurring import jobs and complex data actions.

Document map

Contents

1. Introduction	3
Symptom.....	3
Proof of concept objective	3
2. Proof of Concept architecture	4
3. Solution setup	5
3.1 Create an SAP BDC trial	5
3.2 Install the data product.....	5
3.3 Transform data with SQL Views	6
3.4 Create the planning model	13
3.5 Create the live Actuals view	19
3.6 Connect the live Actuals version	21
3.7 Create the live forecast reference view	23
3.8 Connect the live forecast reference version	27
3.9 Create the SAP Analytics Cloud planning story.....	29
4. Lessons learned and conclusions	34
4.1 Start with the target planning grain	34
4.2 Model compatibility is more important than connector speed.....	34
4.3 Separate live reference data from writable planning data	34
4.4 Use SAP Datasphere for reusable data engineering	34
4.5 Seamless does not mean model-free	34
4.6 Public dimensions can act as lightweight driver stores	34
4.7 Validate performance and security end to end	34
4.8 Conclusion	34
4.9 Recommended next steps	35
5. References and publication notes	35

1. Introduction

Publication note: This document describes a proof of concept implemented in a trial environment. Product behavior, user-interface labels and feature availability may change. Validate the design against current SAP documentation and your tenant configuration before using it in production.

Symptom

A conventional SAC planning implementation would copy actual data into the planning model through imports, then refresh them repeatedly.

It can also require data actions merely to move reference data into the planning area. This creates avoidable data movement, reconciliation effort and operational dependencies.

Proof of concept objective

This proof of concept demonstrates a faster pattern for planning cycles.

- Connect “external actual version” from an existing SAP Datasphere view.
- Create “forecast versions” using SAP Datasphere SQL views, instead of data actions.

Users see current actual data and can enter or calculate forecasts without first importing actuals into the planning model.

Scope note

The guide focuses on the modelling pattern and implementation sequence. Tenant names, object IDs and screenshots are examples from a trial environment and should be replaced with governed production conventions.

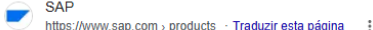
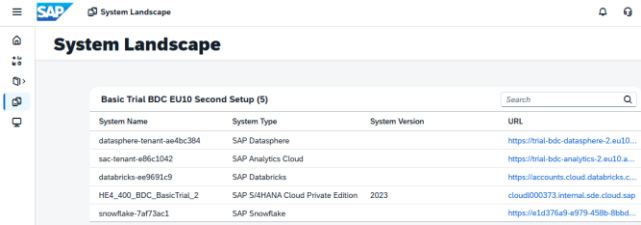
2. Proof of Concept architecture

Retrieve and model actuals → harmonise grain and accounts → virtualise the active version → forecast in SAC → persist plan data in SAP Datasphere.

Shopify sales data	SAP Datasphere	SAP Analytics Cloud	SAP Datasphere storage
Orders, line items and commercial key figures	Ingestion, harmonisation, account/value view and exposed consumption view	Seamless planning model, live actual version and forecast story	Persisted plan facts and reusable planning objects

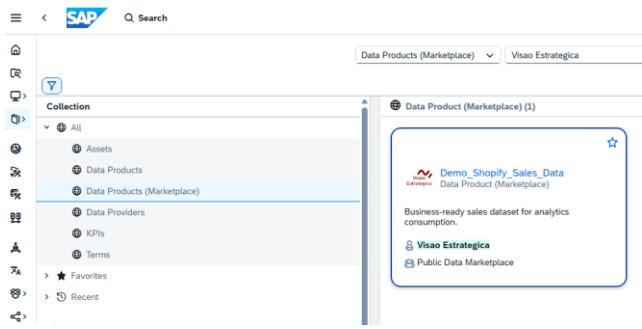
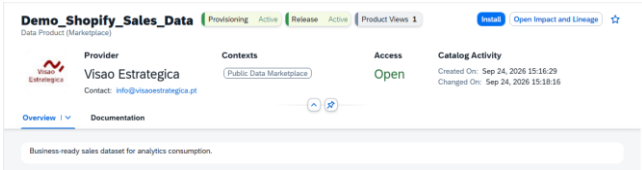
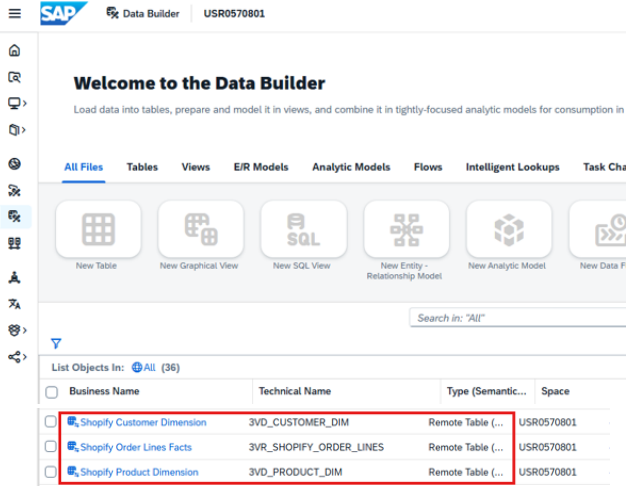
3. Solution setup

3.1 Create an SAP BDC trial

Step	Action	Screenshot																								
-	Create an SAP BDC trial account																									
3.1.1	Start SAP BDC platform free trial	 SAP https://www.sap.com/products · Traduzir esta página · SAP Business Data Cloud trial Learn how to create AI-powered systems that turn intent into execution, October 27–29, 2026. Explore the event ...																								
3.1.2	Under BDC, open System Landscape. You get the access URLs to access SAP Datasphere (data modelling) and SAC (planning and reporting). Select the SAP SAP Datasphere url.	 <table><thead><tr><th>System Name</th><th>System Type</th><th>System Version</th><th>URL</th></tr></thead><tbody><tr><td>datasphere-tenant-ae4bc384</td><td>SAP Datasphere</td><td></td><td>https://trial-bdc-datasphere-2.eu10...</td></tr><tr><td>sac-tenant-e86c1042</td><td>SAP Analytics Cloud</td><td></td><td>https://trial-bdc-analytics-2.eu10.a...</td></tr><tr><td>databricks-e9699c9</td><td>SAP Databricks</td><td></td><td>https://accounts.cloud.databricks.c...</td></tr><tr><td>HE4_A00_BDC_BasicTrial_2</td><td>SAP S/4HANA Cloud Private Edition</td><td>2023</td><td>cloud000373.internal.sde.cloud.sap</td></tr><tr><td>snowflake-7af73ac1</td><td>SAP Snowflake</td><td></td><td>https://s1d376a9-e979-458b-8b8d...</td></tr></tbody></table>	System Name	System Type	System Version	URL	datasphere-tenant-ae4bc384	SAP Datasphere		https://trial-bdc-datasphere-2.eu10...	sac-tenant-e86c1042	SAP Analytics Cloud		https://trial-bdc-analytics-2.eu10.a...	databricks-e9699c9	SAP Databricks		https://accounts.cloud.databricks.c...	HE4_A00_BDC_BasicTrial_2	SAP S/4HANA Cloud Private Edition	2023	cloud000373.internal.sde.cloud.sap	snowflake-7af73ac1	SAP Snowflake		https://s1d376a9-e979-458b-8b8d...
System Name	System Type	System Version	URL																							
datasphere-tenant-ae4bc384	SAP Datasphere		https://trial-bdc-datasphere-2.eu10...																							
sac-tenant-e86c1042	SAP Analytics Cloud		https://trial-bdc-analytics-2.eu10.a...																							
databricks-e9699c9	SAP Databricks		https://accounts.cloud.databricks.c...																							
HE4_A00_BDC_BasicTrial_2	SAP S/4HANA Cloud Private Edition	2023	cloud000373.internal.sde.cloud.sap																							
snowflake-7af73ac1	SAP Snowflake		https://s1d376a9-e979-458b-8b8d...																							

3.2 Install the data product

Step	Action	Screenshot
3.2.1	Under SAP Datasphere, choose “Catalog & Market Place” – Marketplace data products.	 SAP Datasphere Hello, USR0570801 USR0570801 Oct 3, 2025 Auto Refresh Data ON Home Repository Explorer Monitoring Catalog & Marketplace Search Data Product Access Marketplace Data Products Apps Unleash the power of business data

3.2.2	Search for “Demo Shopify Sales Data” provided by “Visao Estrategica”.	 A screenshot of the SAP Data Marketplace interface. The search bar at the top shows 'Demo Shopify Sales Data'. The left sidebar lists various categories like Assets, Data Products, and Data Providers. The main area displays a search result for 'Demo Shopify Sales Data' by 'Visao Estrategica', described as a 'Business-ready sales dataset for analytics consumption'.
3.2.3	Install this data product to get a free set of data to populate your future planning model with actuals.	 A screenshot of the 'Demo Shopify Sales Data' product details page. It shows the provider 'Visao Estrategica', the context 'Public Data Marketplace', and the access status 'Open'. The page also includes a 'Catalog Activity' section with creation and change timestamps.
3.2.4	After installation, three new remote tables are available in SAP Datasphere “data builder” area.	 A screenshot of the SAP Data Builder interface. The 'Welcome to the Data Builder' message is at the top. Below it, there are tabs for 'All Files', 'Tables', 'Views', etc. The 'Tables' tab is selected, showing a list of objects. Three tables are listed: '3VD_Customer Dimension', '3VR_SHOPIFY_ORDER_LINES', and '3VD_PRODUCT_DIM'. These three tables are highlighted with a red box, indicating they are the new remote tables available after installation.

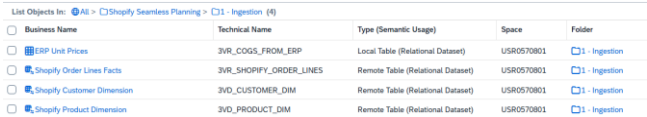
At this point, you have access to Actual Shopify Sales Orders, Products and Customers

3.3 Transform data with SQL Views

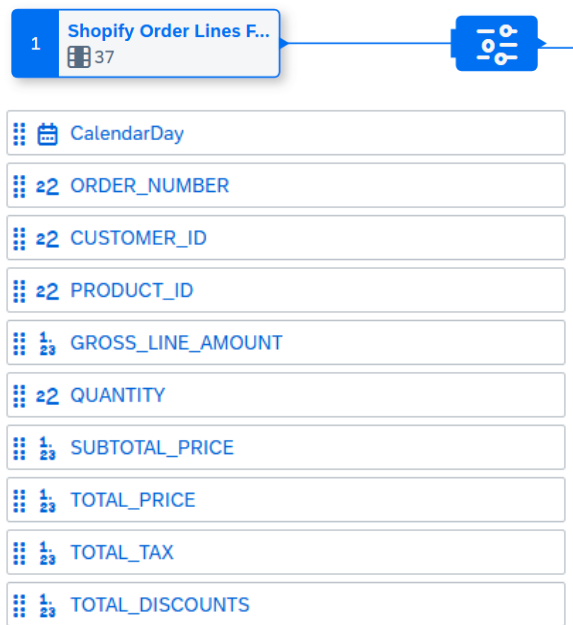
After installing Data Product view you need to create auxiliar transformation to consume data with correct level of granularity in SAP Planning model

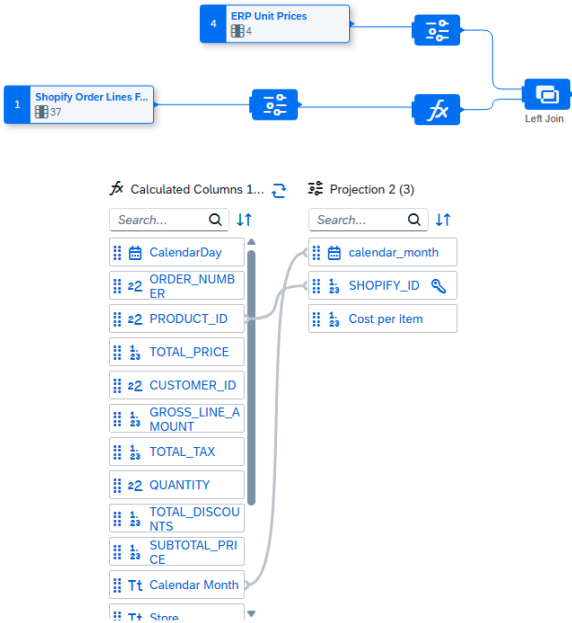
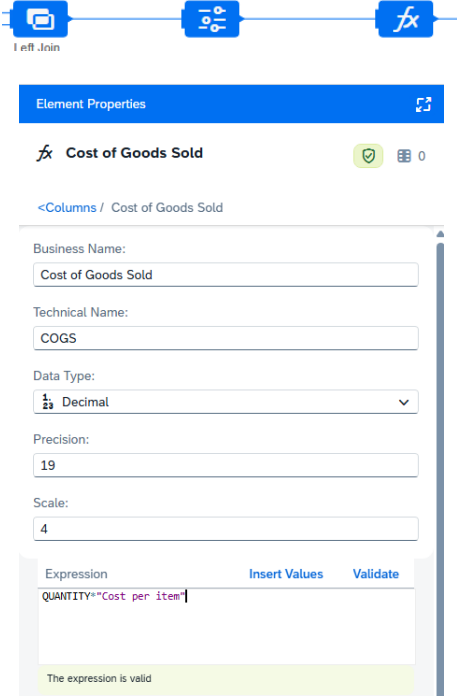
Step	Action	Screenshot
------	--------	------------

3.3.1	Go to SAP Datasphere Data Builder and create a proper structure of folders to organise your data layers.	▼ List Objects In: 🌐 All > 📁 Shopify Seamless Planning (5) <table><tr><th>☐ Business Name</th><th>Technical Name</th><th>Type (Semantic Usage)</th></tr><tr><td>☒ 1 - Ingestion</td><td>–</td><td>Folder</td></tr><tr><td>☐ 2 - Harmonization</td><td>–</td><td>Folder</td></tr><tr><td>☐ 3 - Propagation</td><td>–</td><td>Folder</td></tr><tr><td>☐ 4 - Analytics</td><td>–</td><td>Folder</td></tr><tr><td>☐ 5 - SAC Planning</td><td>–</td><td>Folder</td></tr></table>	☐ Business Name	Technical Name	Type (Semantic Usage)	☒ 1 - Ingestion	–	Folder	☐ 2 - Harmonization	–	Folder	☐ 3 - Propagation	–	Folder	☐ 4 - Analytics	–	Folder	☐ 5 - SAC Planning	–	Folder																																														
☐ Business Name	Technical Name	Type (Semantic Usage)																																																																
☒ 1 - Ingestion	–	Folder																																																																
☐ 2 - Harmonization	–	Folder																																																																
☐ 3 - Propagation	–	Folder																																																																
☐ 4 - Analytics	–	Folder																																																																
☐ 5 - SAC Planning	–	Folder																																																																
3.3.2	Move your data products remote tables into “1 – Ingestion” folder This data product provides you Facts and Dimensions (Customer, Product)	List Objects In: 🌐 All > 📁 Shopify Seamless Planning > 📁 1 - Ingestion (4) <table><tr><th>☐ Business Name</th><th>Technical Name</th><th>Type (Semantic Usage)</th><th>Space</th><th>Folder</th></tr><tr><td>☒ Shopify Product Dimension</td><td>3VD_PRODUCT_DIM</td><td>Remote Table (Relational Dataset)</td><td>USR0570801</td><td>📁 1 - Ingestion</td></tr><tr><td>☒ Shopify Order Lines Facts</td><td>3VR_SHOPIFY_ORDER_LINES</td><td>Remote Table (Relational Dataset)</td><td>USR0570801</td><td>📁 1 - Ingestion</td></tr><tr><td>☒ Shopify Customer Dimension</td><td>3VD_CUSTOMER_DIM</td><td>Remote Table (Relational Dataset)</td><td>USR0570801</td><td>📁 1 - Ingestion</td></tr></table>	☐ Business Name	Technical Name	Type (Semantic Usage)	Space	Folder	☒ Shopify Product Dimension	3VD_PRODUCT_DIM	Remote Table (Relational Dataset)	USR0570801	📁 1 - Ingestion	☒ Shopify Order Lines Facts	3VR_SHOPIFY_ORDER_LINES	Remote Table (Relational Dataset)	USR0570801	📁 1 - Ingestion	☒ Shopify Customer Dimension	3VD_CUSTOMER_DIM	Remote Table (Relational Dataset)	USR0570801	📁 1 - Ingestion																																												
☐ Business Name	Technical Name	Type (Semantic Usage)	Space	Folder																																																														
☒ Shopify Product Dimension	3VD_PRODUCT_DIM	Remote Table (Relational Dataset)	USR0570801	📁 1 - Ingestion																																																														
☒ Shopify Order Lines Facts	3VR_SHOPIFY_ORDER_LINES	Remote Table (Relational Dataset)	USR0570801	📁 1 - Ingestion																																																														
☒ Shopify Customer Dimension	3VD_CUSTOMER_DIM	Remote Table (Relational Dataset)	USR0570801	📁 1 - Ingestion																																																														
3.3.3	Shopify APIs does not provide historical cost of goods sold, that usually comes from ERP (SAP). To address this missing KPI, we create a demo table and import COGS from flat file. Use a CSV with simple structure.	<table><tr><th>calendar_month</th><th>SHOPIFY_ID</th><th>Cost per item</th><th>Inventory quantity</th></tr><tr><td>202609</td><td>16,097,152,205,182.00</td><td>9.27</td><td>22</td></tr><tr><td>202609</td><td>16,097,152,237,950.00</td><td>12.6</td><td>29</td></tr><tr><td>202609</td><td>16,097,152,303,486.00</td><td>15.94</td><td>36</td></tr><tr><td>202609</td><td>16,097,152,336,254.00</td><td>19.27</td><td>43</td></tr><tr><td>202609</td><td>16,097,152,401,790.00</td><td>22.61</td><td>50</td></tr><tr><td>202609</td><td>16,097,152,434,558.00</td><td>25.94</td><td>57</td></tr><tr><td>202609</td><td>16,097,152,467,326.00</td><td>29.28</td><td>64</td></tr><tr><td>202609</td><td>16,097,152,500,094.00</td><td>32.61</td><td>71</td></tr><tr><td>202609</td><td>16,097,152,532,862.00</td><td>35.95</td><td>78</td></tr><tr><td>202609</td><td>16,097,152,565,630.00</td><td>39.28</td><td>85</td></tr><tr><td>202609</td><td>16,097,152,598,398.00</td><td>42.62</td><td>92</td></tr><tr><td>202609</td><td>16,097,152,631,166.00</td><td>45.95</td><td>19</td></tr><tr><td>202609</td><td>16,097,152,696,702.00</td><td>49.29</td><td>26</td></tr><tr><td>202609</td><td>16,097,152,729,470.00</td><td>52.62</td><td>33</td></tr><tr><td>202609</td><td>16,097,152,795,006.00</td><td>55.96</td><td>40</td></tr></table>	calendar_month	SHOPIFY_ID	Cost per item	Inventory quantity	202609	16,097,152,205,182.00	9.27	22	202609	16,097,152,237,950.00	12.6	29	202609	16,097,152,303,486.00	15.94	36	202609	16,097,152,336,254.00	19.27	43	202609	16,097,152,401,790.00	22.61	50	202609	16,097,152,434,558.00	25.94	57	202609	16,097,152,467,326.00	29.28	64	202609	16,097,152,500,094.00	32.61	71	202609	16,097,152,532,862.00	35.95	78	202609	16,097,152,565,630.00	39.28	85	202609	16,097,152,598,398.00	42.62	92	202609	16,097,152,631,166.00	45.95	19	202609	16,097,152,696,702.00	49.29	26	202609	16,097,152,729,470.00	52.62	33	202609	16,097,152,795,006.00	55.96	40
calendar_month	SHOPIFY_ID	Cost per item	Inventory quantity																																																															
202609	16,097,152,205,182.00	9.27	22																																																															
202609	16,097,152,237,950.00	12.6	29																																																															
202609	16,097,152,303,486.00	15.94	36																																																															
202609	16,097,152,336,254.00	19.27	43																																																															
202609	16,097,152,401,790.00	22.61	50																																																															
202609	16,097,152,434,558.00	25.94	57																																																															
202609	16,097,152,467,326.00	29.28	64																																																															
202609	16,097,152,500,094.00	32.61	71																																																															
202609	16,097,152,532,862.00	35.95	78																																																															
202609	16,097,152,565,630.00	39.28	85																																																															
202609	16,097,152,598,398.00	42.62	92																																																															
202609	16,097,152,631,166.00	45.95	19																																																															
202609	16,097,152,696,702.00	49.29	26																																																															
202609	16,097,152,729,470.00	52.62	33																																																															
202609	16,097,152,795,006.00	55.96	40																																																															
3.3.4	Shopify APIs will not give you correct Cost of Goods Sold, that usually comes from ERP (SAP). To address this missing data we create a demo table and import COGS from flat file. Name the table “3VR_COGS_FROM_ERP”	+ ▼ 🔗 ✎ 🗑 🔄 🔗 📁 📄 🔊 ⬆ ⬆ ⚙ 📊 ☰ Import CSV File Import Objects from CSN/JSON File Import Entities Import Remote Tables Validate Remote Tables Import Permissions Business Name: ERP Unit Prices Technical Name: 3VR_COGS_FROM_ERP Semantic Usage: Relational Dataset Columns (4) <table><tr><th>☐ Business Name</th><th>Technical Name</th><th>Data Type</th></tr><tr><td>☐ calendar_month</td><td>calendar_month</td><td>Date</td></tr><tr><td>☒ SHOPIFY_ID</td><td>SHOPIFY_ID</td><td>Decimal(38, 18)</td></tr><tr><td>☐ Cost per item</td><td>Cost per item</td><td>Decimal(38, 18)</td></tr><tr><td>☐ Inventory quantity</td><td>Inventory quantity</td><td>Integer64</td></tr></table>	☐ Business Name	Technical Name	Data Type	☐ calendar_month	calendar_month	Date	☒ SHOPIFY_ID	SHOPIFY_ID	Decimal(38, 18)	☐ Cost per item	Cost per item	Decimal(38, 18)	☐ Inventory quantity	Inventory quantity	Integer64																																																	
☐ Business Name	Technical Name	Data Type																																																																
☐ calendar_month	calendar_month	Date																																																																
☒ SHOPIFY_ID	SHOPIFY_ID	Decimal(38, 18)																																																																
☐ Cost per item	Cost per item	Decimal(38, 18)																																																																
☐ Inventory quantity	Inventory quantity	Integer64																																																																

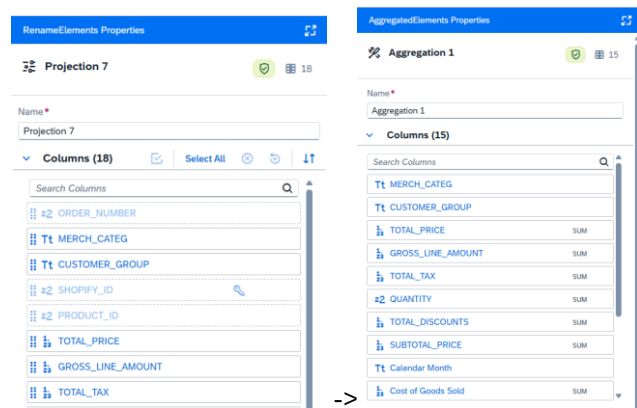
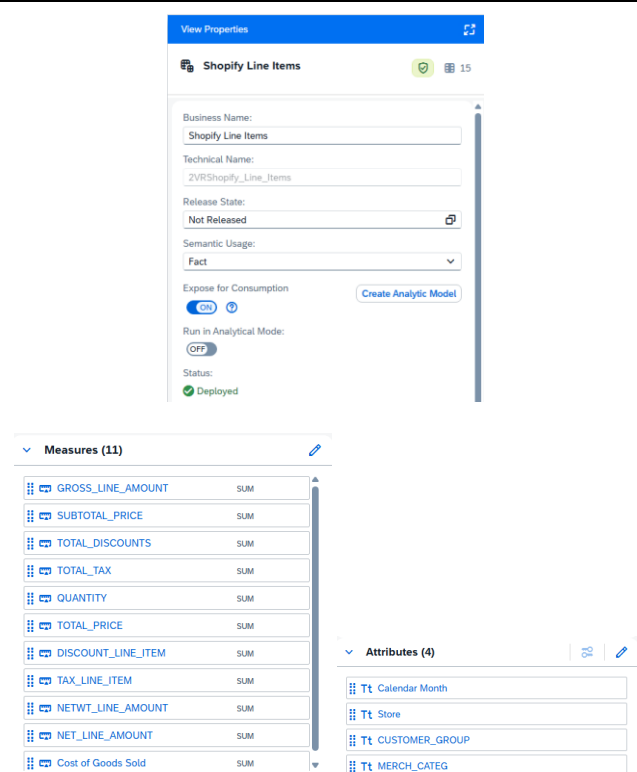
3.3.5	Your Data Foundation layer is ready and contains four tables.	
-------	---	--

At this point, you have access to Actual Inventory Unit Prices, relevant for COGS calculation

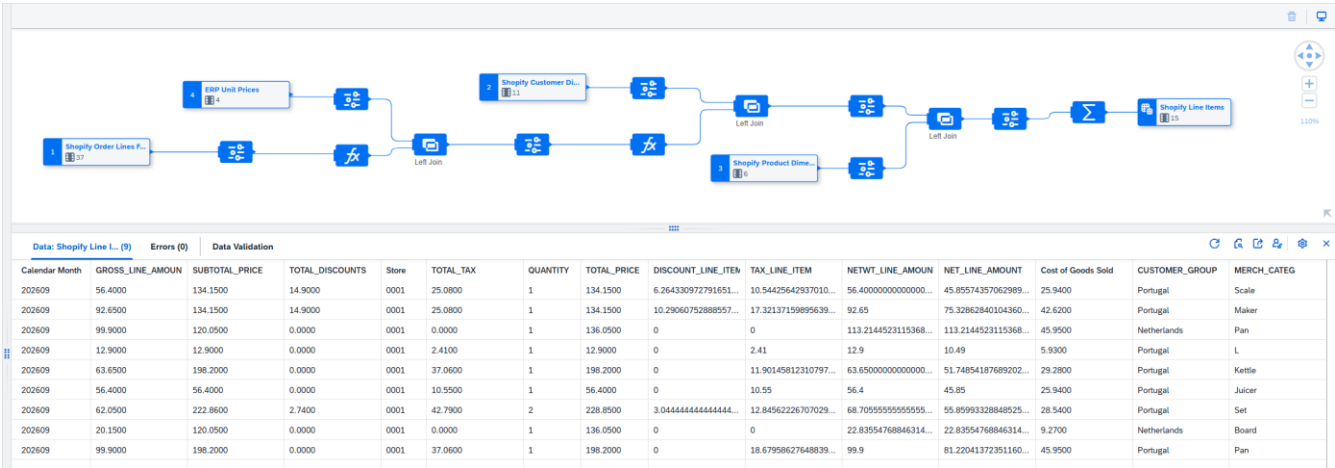
Step	Action	Screenshot
3.3.6	Go to folder “2 – Harmonisation”. Start creating new Graphical view “3VR_SHOPIFY_ORDER_LINES”	
3.3.7	Add Calculated columns: Calendar Month (String 6) TO_NVARCHAR(CalendarDay, 'YYYYMM') Store (String 4) '0001' DISCOUNT_LINE_ITEM (Decimal 34) $TOTAL_DISCOUNTS * (GROSS_LINE_AMOUNT / SUBTOTAL_PRICE)$ TAX_LINE_ITEM (Decimal 34) $TOTAL_TAX * (GROSS_LINE_AMOUNT / SUBTOTAL_PRICE)$ NETWT_LINE_AMOUNT (Decimal 34) $TOTAL_PRICE * (GROSS_LINE_AMOUNT / SUBTOTAL_PRICE)$ NET_LINE_AMOUNT $(TOTAL_PRICE - TOTAL_TAX) * (GROSS_LINE_AMOUNT / SUBTOTAL_PRICE)$	

3.3.8	Join Shopify sales with ERP Unit Prices to get “Cost per item”	
3.3.9	Add Calculated columns: COGS (Decimal 19,4) QUANTITY*"Cost per item"	
3.3.10	Join your data with 3VD_CUSTOMER_DIM (Customer Dimension) We want to aggregate actual sales by Customer Groups.	

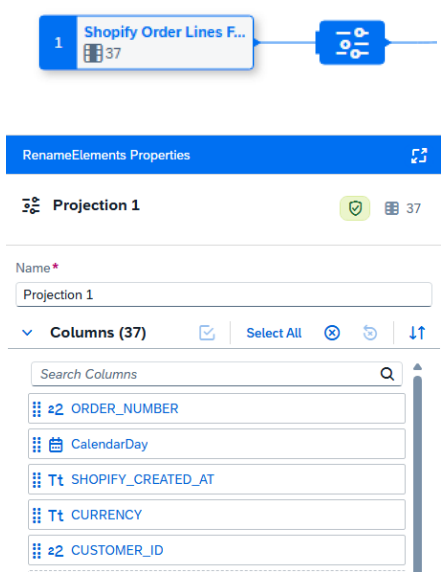
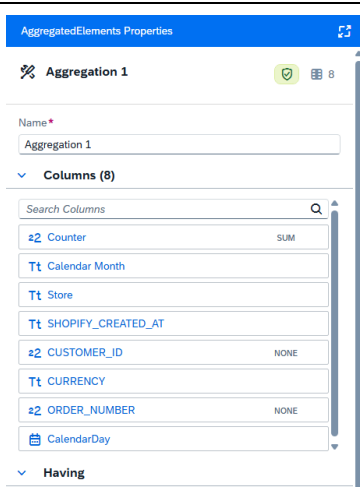
		▼ Mappings AllMappedUnmapped ✂Calculated Columns 2...↺↻Projection 4 (2) Search...Q↓↑ Search...Q↓↑ ORDER_NUMBER Cost per item PRODUCT_ID TOTAL_PRICE CUSTOMER_ID GROSS_LINE_A MOUNT TOTAL_TAX QUANTITY TOTAL_DISCOU NTS SUBTOTAL_PRI CE Calendar Month Store TtCUSTOMER_GRO UP 22SHOPIFY_ID
3.3.11	Join your data with 3VD_PRODUCT_DIM (Product Dimension) We want to aggregate actual sales by Merchandise Categories.	Left Join Left Join 3Shopify Product Dime...6 ▼ Mappings AllMappedUnmapped ✂Projection 5 (16)↺↻Projection 6 (2) Search...Q↓↑ Search...Q↓↑ ORDER_NUMBER TtCUSTOMER_G ROUP PRODUCT_ID TOTAL_PRICE GROSS_LINE_A MOUNT TOTAL_TAX QUANTITY TOTAL_DISCOU NTS SUBTOTAL_PRI CE Calendar Month Store DISCOUNT_LIN TtMERCH_CATEG 22SHOPIFY_ID
3.3.12	Aggregate facts by: - Calendar Month - Store - Customer Group - Merchandise Categ.	Σ

3.3.13	First, exclude detailed dimensions such as order id shopify id and product id. Then, aggregate (SUM) metrics.	
3.3.14	Finally define the metadata of your graphical view. - fact table - 11 measures - four attributes As this view will be used only for Planning we don't need to define additional semantic information.	

“Shopify Line Items” result:



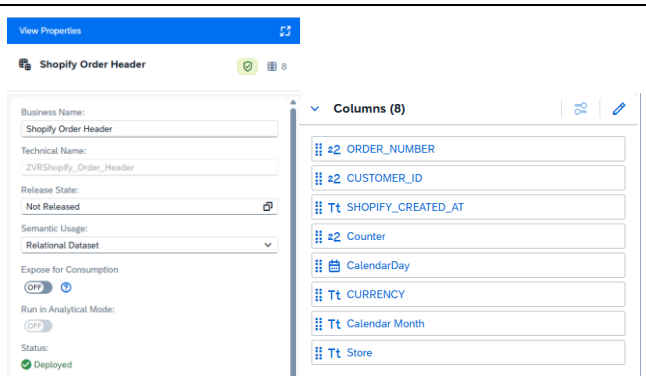
Calendar Month	GROSS_LINE_AMOUNT	SUBTOTAL_PRICE	TOTAL_DISCOUNTS	Store	TOTAL_TAX	QUANTITY	TOTAL_PRICE	DISCOUNT_LINE_ITEM	TAX_LINE_ITEM	NETWT_LINE_AMOUNT	NET_LINE_AMOUNT	Cost of Goods Sold	CUSTOMER_GROUP	MERCH_CATEG
202609	56.4000	134.1500	14.9000	0001	25.0800	1	134.1500	6.264330972791651...	10.54425642937010...	56.40000000000000...	45.85574357062989...	25.9400	Portugal	Scale
202609	92.6500	134.1500	14.9000	0001	25.0800	1	134.1500	10.290607528889557...	17.32137159895639...	92.65	75.32862840104360...	42.6200	Portugal	Maker
202609	99.9000	120.0500	0.0000	0001	0.0000	1	136.0500	0	0	113.2144523115368...	113.2144523115368...	45.9500	Netherlands	Pan
202609	12.9000	12.9000	0.0000	0001	2.4100	1	12.9000	0	2.41	12.9	10.49	5.9300	Portugal	L
202609	63.6500	198.2000	0.0000	0001	37.0600	1	198.2000	0	11.90145812310797...	63.65000000000000...	51.74854187689202...	29.2800	Portugal	Kettle
202609	56.4000	56.4000	0.0000	0001	10.5500	1	56.4000	0	10.55	56.4	45.85	25.9400	Portugal	Juicer
202609	62.0500	222.8600	2.7400	0001	42.7900	2	228.8500	3.044444444444444...	12.84562226707029...	68.70555555555555...	55.85993328848525...	28.5400	Portugal	Set
202609	20.1500	120.0500	0.0000	0001	0.0000	1	136.0500	0	0	22.83554768846314...	22.83554768846314...	9.2700	Netherlands	Board
202609	99.9000	198.2000	0.0000	0001	37.0600	1	198.2000	0	18.67598627648839...	99.9	81.22041372351160...	45.9500	Portugal	Pan

Step	Action	Screenshot
3.3.15	Go to folder “2 – Harmonisation”. Start creating new Graphical view “2VRShopify_Order_Header” Create a projection only with relevant fields (5). Note: the purpose of this auxiliary view will be to count new customers.	 The screenshot shows the 'RenameElements Properties' window for 'Projection 1'. The 'Name' field is 'Projection 1'. Under 'Columns (37)', a search bar is present, and five columns are selected: ORDER_NUMBER (2), CalendarDay, SHOPIFY_CREATED_AT (T), CURRENCY (T), and CUSTOMER_ID (2).
3.3.16	Add Calculated columns: Calendar Month (String 6) TO_NVARCHAR(CalendarDay, 'YYYYMM') Store (String 4) '0001' Counter (Integer) (items counter) 1	 The screenshot shows a data flow diagram with a calculated column function (fx) connected to a data source.
3.3.17	Aggregate facts by: - Calendar Month - Store - Shopify Created at - Customer Id - Currency - Order Number - Calendar Day	 The screenshot shows the 'AggregatedElements Properties' window for 'Aggregation 1'. The 'Name' field is 'Aggregation 1'. Under 'Columns (8)', a search bar is present, and five columns are selected: Counter (2), Calendar Month (T), Store (T), SHOPIFY_CREATED_AT (T), and CUSTOMER_ID (2). The 'Having' section is also visible.

3.3.18

Finally define the metadata of your graphical view.

- relational dataset



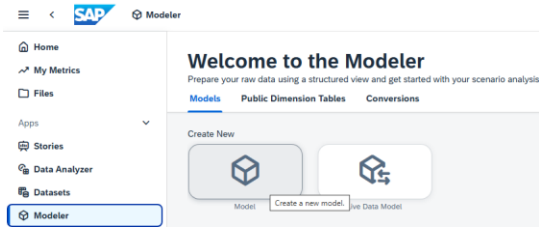
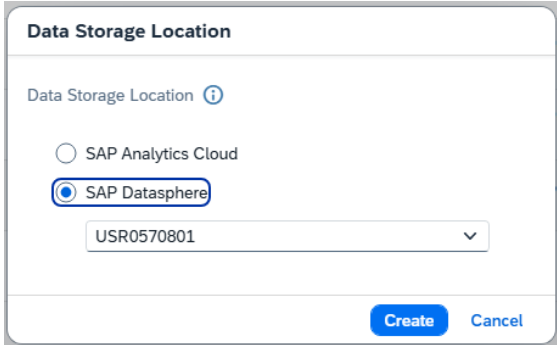
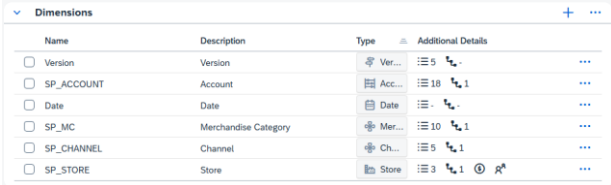
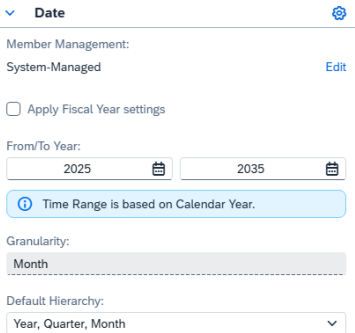
“Shopify order-header” result:

ORDER_NUMBER	CUSTOMER_ID	SHOPIFY_CREATED_AT	Counter	CalendarDay	CURRENCY	Calendar Month	Store
1.001	25,713,185,358,206	2026-09-22T15:09:55+01:00	3	Sep 22, 2026	EUR	202609	0001
1.002	25,713,184,801,150	2026-09-27T17:42:29+01:00	2	Sep 27, 2026	EUR	202609	0001
1.003	25,713,184,899,454	2026-09-27T17:45:43+01:00	2	Sep 27, 2026	EUR	202609	0001
1.004	25,713,185,390,974	2026-09-27T19:29:07+01:00	1	Sep 27, 2026	EUR	202609	0001
1.005	25,713,185,358,206	2026-09-28T11:36:07+01:00	1	Sep 28, 2026	EUR	202609	0001
1.006	25,713,185,358,206	2026-09-28T11:37:44+01:00	1	Sep 28, 2026	EUR	202609	0001

At this step, you have defined proper order-item and order-header Views.

3.4 Create the planning model

Step	Action	Screenshot
3.4.1	In this section we will create the planning model in SAC	
3.4.2	Under BDC, open System Landscape. You get the access URLs to access SAP Datasphere (data modelling) and SAC (planning and reporting). Select the SAP Analytics Cloud url.	

3.4.3	Navigate to Modeler, Select Create New Model Choose Start with an empty Model Select Next	
3.4.4	Choose SAP Datasphere, as your data storage location, and click on Create This will also make planning data immediately available for further data builder developments.	
3.4.5	Define 6 dimensions: - Version - Account - Date - Merchandise Category - Channel - Store	
3.4.6	Edit “Date” dimension to restrict time interval and define month granularity	

3.4.7

Create Account type, Private SP_ACCOUNT
Dimension

Edit account to add “H01” Parent-child
hierarchy

Note: this is a demo, for real scenarios
Public dimensions may be more appropriate

Dimension Table

Account
SP_ACCOUNT

General

Type:
Account

Name:
SP_ACCOUNT

Description:
Account

Data Storage (SAP Datasphere)

SAP Datasphere Space:
USR0570801

Rights / Access

☐ Data Access Control
☐ Data Locking Ownership

Properties

There are currently no properties associated with the dimension. Add currency, unit, geo or generic properties to enrich the dimension. Each property you add has an associated column in the dimension.

+ Create Property

Hierarchies

Hierarchy
Parent-Child Hierarchy

3.4.8

Edit Account dimensions to define the
following member IDs

Member ID	Description	Formula	Account Type	Agg.	Exception Agg.	Exception	Required	Scale	Decimal	Unit Type	Fixed Unit	Threshold	HOW
1	AD001	Gross Sales (with Tax)	H001							Currency			VS000
2	AD002	Discounts (with Tax)	H002							Currency			VS000
3	AD003	Tot	H001							Currency			VS000
4	AD004	COGS	H002							Currency			VS000
5	AD005	Other Variable Expenses	H003							Currency			VS000
6	DD001	Gross Sales Variance %	++000			Average excl. NULL	Date		Percent	1			VS000
7	DD002	Discount Variance %	++000			Average excl. NULL	Date		Percent	1			VS000
8	DD003	Active Customers Variat.	++000			Average excl. NULL	Date		Percent	1			VS000
9	DD004	New Customers Variat.	++000			Average excl. NULL	Date		Percent	1			VS000
10	H001	New Sales	H001	[AD001-AD002-AD003]						Currency			VS000
11	H002	Gross Margin	H002	[AD003-AD004]						Currency			VS000
12	H003	Commercial Margin	H003	[AD003-AD005]						Currency			VS000
13	SD000	Statistical Key Figures	++000	0					0	Name			VS000
14	SD001	Active Customers	SD000						0	Name			VS000
15	SD002	New Customers	SD000						0	Name			VS000

** See detail below

3.4.9

Create Generic type, Private SP_MC
Dimension

Edit Merchandise Categ. dimension to add
“H01” Parent-child hierarchy

Model

Dimension

Merchandise Category
SP_MC

General

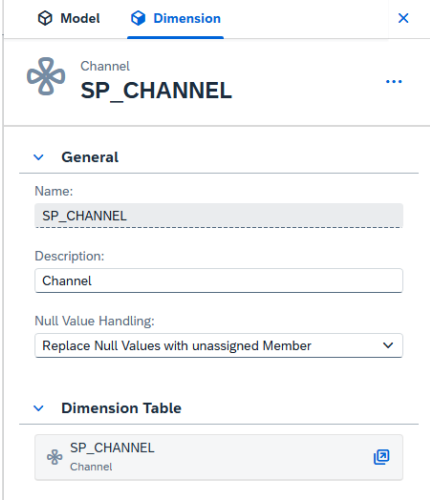
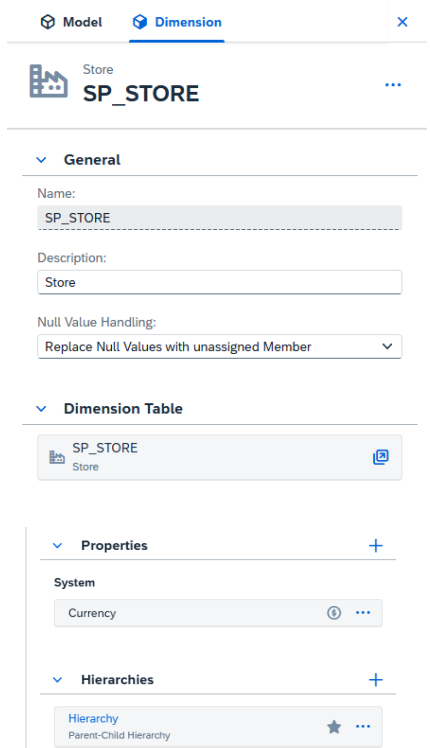
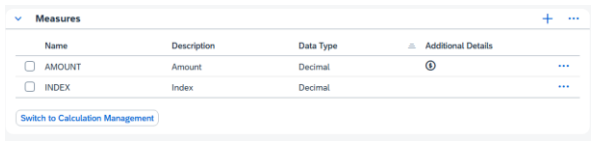
Name:
SP_MC

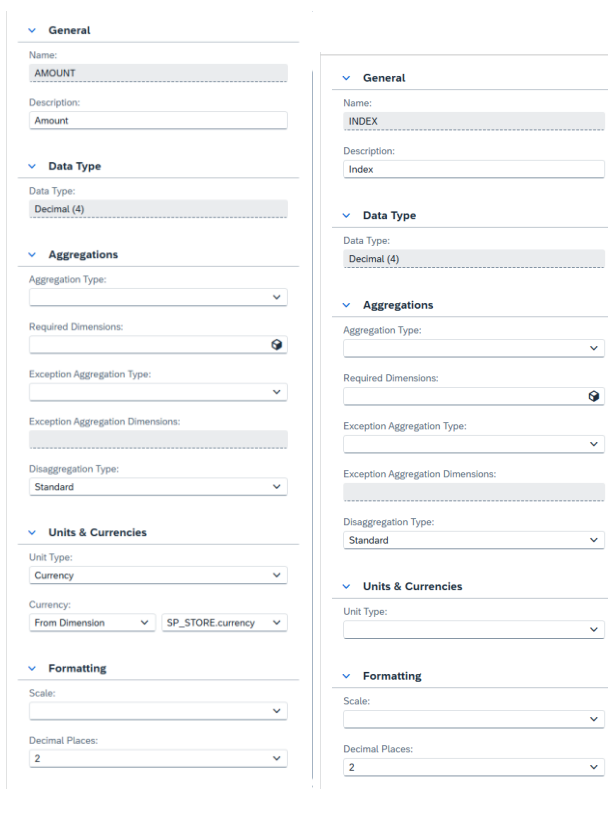
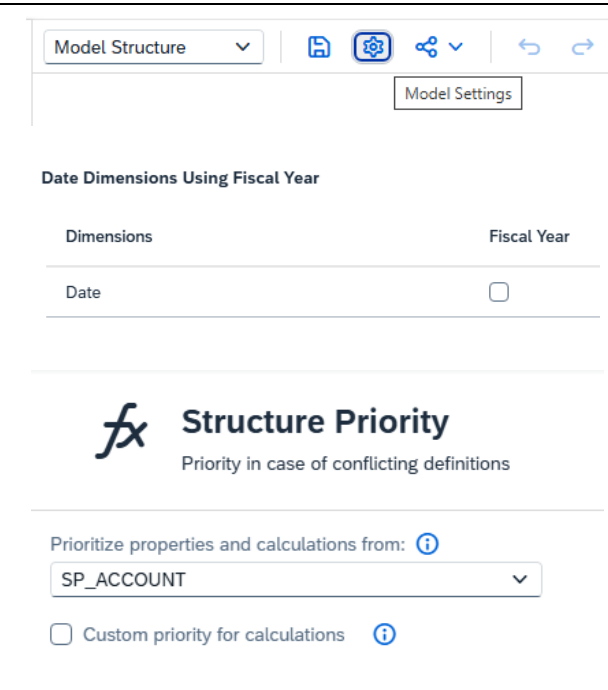
Description:
Merchandise Category

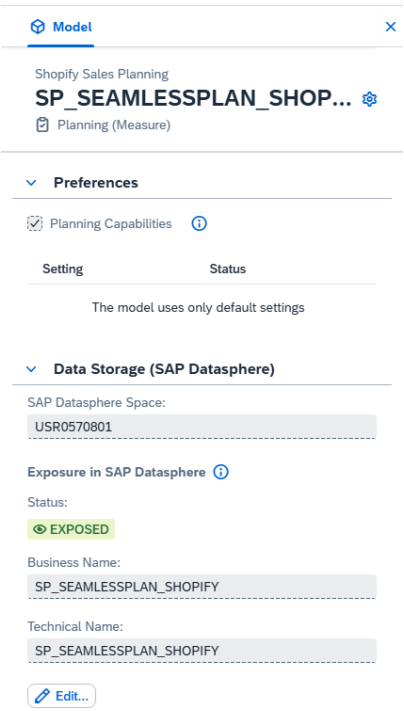
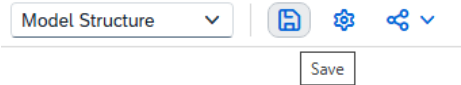
Null Value Handling:
Replace Null Values with unassigned Member

Dimension Table

SP_MC
Merchandise Category

3.4.10	Create Generic type, Private SP_CHANNEL Dimension Edit Channel dimensions to add “H01” Parent-child hierarchy	
3.4.11	Create Organization type, Private SP_STORE Dimension Edit Store dimensions to add “H01” Parent-child hierarchy. Confirm Organization dimensions comes with Currency attribute by default.	
3.4.12	Define 2 measures: - “Amount” for Financial indicators and using currency - “Index” for other Statistical Key Figures, not currency related.	

	
3.4.13 Go to Model Settings. Under Date Settings, disable Fiscal Year usage. Under Structure Priority, choose “SP_Account” to take priority in calculation. This is very important for Accounts formulas and exception aggregation usage.	

3.4.14	Expose fact table in SAP Datasphere. Note: not mandatory for this purpose, but beneficial for additional reporting needs.	
3.4.15	Take this time to save your planning model. Name: SP_SEAMLESSPLAN_SHOPIFY Description: Shopify Sales Planning	

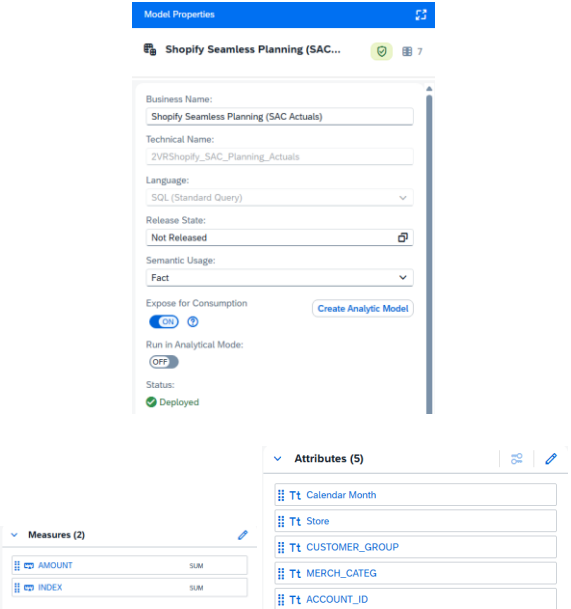
Accounting

Id	Description	Hier	Formula	Acc Tp	Exc Aggr	Except Dim	Scale	Dec	Unit
A001	Gross Sales (with Tax)	H001		INC					CUR
A002	Discounts (with Tax)	H001		EXP					CUR
A003	Tax	H001		EXP					CUR
A004	COGS	H002		EXP					CUR
A005	Other Variable Expenses	H003		EXP					CUR
D001	Gross Sales Variance %	<root>			AVERAGENULL	Date	PERCENT	1	
D002	Discount Variance %	<root>			AVERAGENULL	Date	PERCENT	1	
D003	Active Customers Variance %	<root>			AVERAGENULL	Date	PERCENT	1	
D004	New Customers Variance %	<root>			AVERAGENULL	Date	PERCENT	1	
H001	Net Sales	H002	[A001]+[A002]+[A003]	INC					CUR
H002	Gross Margin	H003	[H001]+[A004]	INC					CUR
H003	Commercial Margin	<root>	[H002]+[A005]	INC					CUR
S000	Statistical Key Figures	<root>	0	NFIN				0	NON
S001	Active Customers	S000		NFIN				0	NON
S002	New Customers	S000		NFIN				0	NON

At this step, you define a planning model. You don't have data yet.

3.5 Create the live Actuals view

Step	Action	Screenshot
3.5.1	Go to folder “2 – Harmonisation”. Start creating new SQL view “2VRShopify_SAC_Planning_Actuals” We will transform previous views to an “Account Model structure” facilitating integration in SAC Planning. This view will provide actuals for a planning model with following accounts: -- A001 Gross sales with Tax -- A002 Discounts with tax -- A003 Taxes -- A004 Cost of Goods Sold -- S001 Active Customers -- S002 New Customers	<pre>-- A001 Gross sales with Tax SELECT "Calendar_Month", "Store", "CUSTOMER_GROUP", "MERCH_CATEG", 'A001' AS "ACCOUNT_ID", CAST(("GROSS_LINE_AMOUNT" AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" UNION ALL -- A002 Discounts with tax SELECT "Calendar_Month", "Store", "CUSTOMER_GROUP", "MERCH_CATEG", 'A002' AS "ACCOUNT_ID", CAST(("DISCOUNT_LINE_ITEM" AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" UNION ALL -- A003 Taxes SELECT "Calendar_Month", "Store", "CUSTOMER_GROUP", "MERCH_CATEG", 'A003' AS "ACCOUNT_ID", CAST(("TAX_LINE_ITEM" AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" UNION ALL -- A004 Cost of Goods Sold SELECT "Calendar_Month", "Store", "CUSTOMER_GROUP", "MERCH_CATEG", 'A004' AS "ACCOUNT_ID", CAST(("COGS" AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" UNION ALL -- S001 Active Customers SELECT "Calendar_Month", "Store", '#', '#', 'S001' AS "ACCOUNT_ID", CAST(0 AS DECIMAL(15,4)) AS "AMOUNT", CAST(COUNT(DISTINCT "CUSTOMER_ID") AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Order_Header" GROUP BY "Calendar_Month", "Store" UNION ALL -- S002 New Customers SELECT T."Calendar_Month", T."Store", '#', '#', 'S002', CAST(0 AS DECIMAL(15,4)) AS "AMOUNT",</pre>

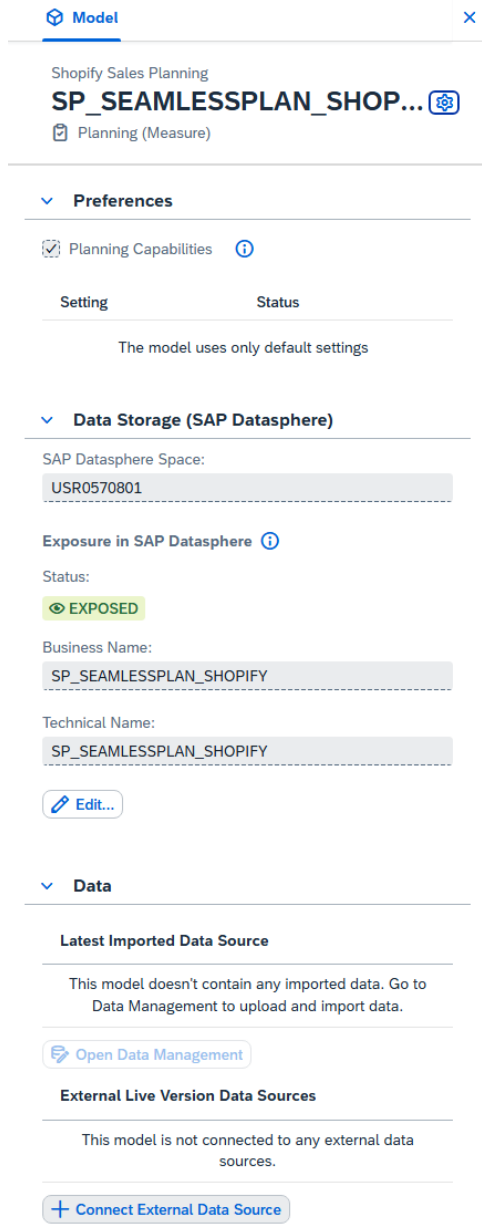
		<pre>CAST(SUM(NEW_CUSTOMER) AS DECIMAL(15,4)) AS "INDEX" FROM (SELECT "Calendar_Month", "Store", CASE WHEN LAG("ORDER_NUMBER") OVER (PARTITION BY "CUSTOMER_ID" ORDER BY "SHOPIFY_CREATED_AT", "ORDER_NUMBER") IS NULL THEN 1 ELSE 0 END AS NEW_CUSTOMER FROM "2VRShopify_Order_Header") T GROUP BY "Calendar_Month", "Store"</pre>
3.5.2	Finally define the metadata of your graphical view. - fact table - two measures - five attributes This is Planning model structure	

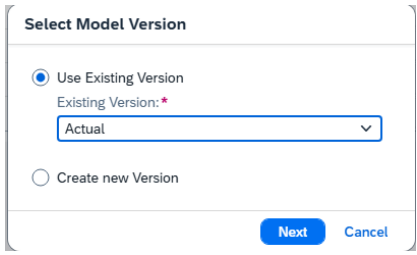
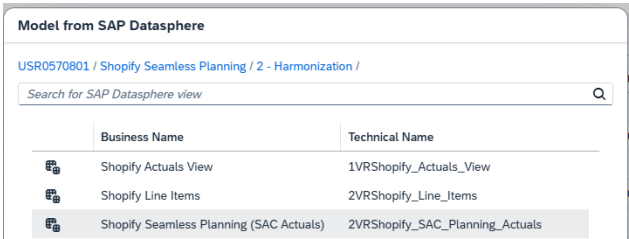
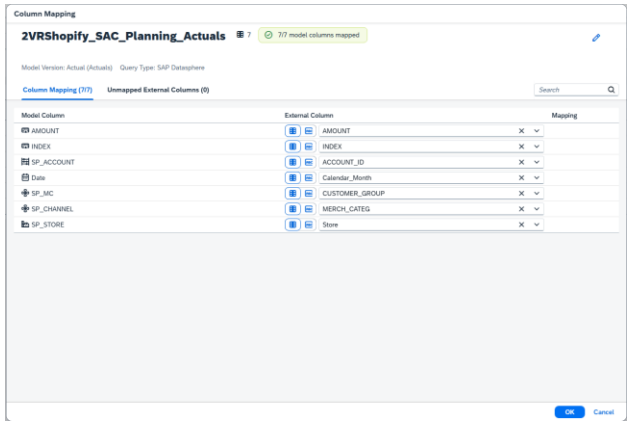
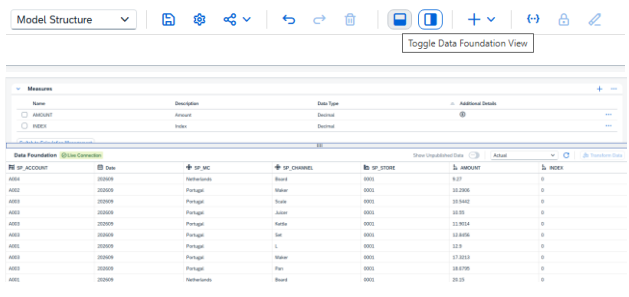
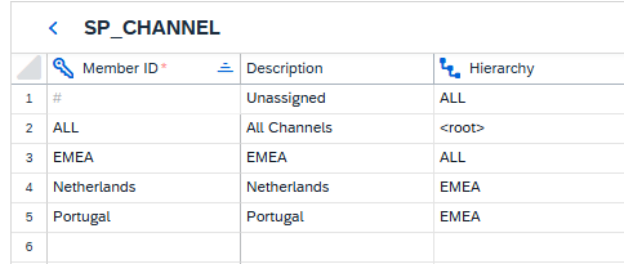
“Shopify Seamless Planning (SAC Actuals)” result:

Data: Shopify Seaml... (38)							Errors (0)	Data Validation
Calendar Month	Store	CUSTOMER_GROUP	MERCH_CATEG	ACCOUNT_ID	AMOUNT	INDEX		
202609	0001	Portugal	Pan	A001	99.9000	0.0000		
202609	0001	Portugal	Set	A001	62.0500	0.0000		
202609	0001	Netherlands	Board	A001	20.1500	0.0000		
202609	0001	Netherlands	Pan	A001	99.9000	0.0000		
202609	0001	Portugal	Maker	A001	92.6500	0.0000		
202609	0001	Portugal	L	A001	12.9000	0.0000		
202609	0001	Portugal	Kettle	A001	63.6500	0.0000		
202609	0001	Portugal	Scale	A001	56.4000	0.0000		
202609	0001	Portugal	Juicer	A001	56.4000	0.0000		
202609	0001	Portugal	Pan	A002	0.0000	0.0000		

At this step, you create proper sql view to deliver live actual data to planning model.

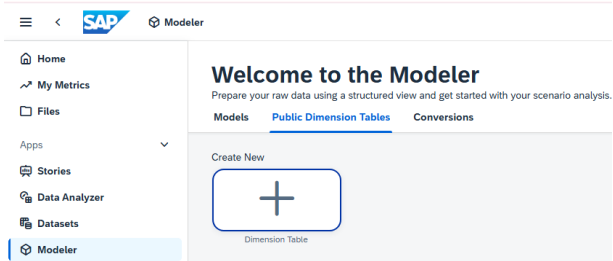
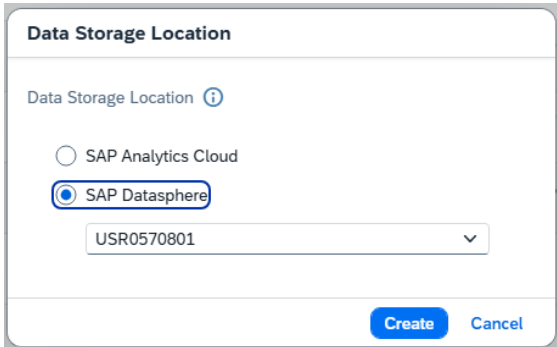
3.6 Connect the live Actuals version

Step	Action	Screenshot
3.6.1	Finally, we want to use Seamless Planning advantages. Under right side Details panel, choose Model Settings	
3.6.2	First, we are going to connect an external live version datasource to Version “0” Actuals. Select Connect External Data Source	

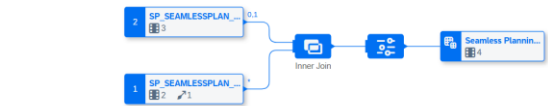
3.6.3	Use existing version “Actual”	
3.6.4	Choose your SAP Datasphere View to integrate a Live Actual version in your planning model.	
3.6.5	Map SAP Datasphere fields to Planning model Dimensions and Measures. Select OK Note: Once you click OK, you will get errors because of missing dimensions ID's. You can accept to create Member Id's at this step and the errors disappear.	
3.6.6	“Toggle Data Foundation View” to display Live Actual data.	
3.6.7	Edit dimensions to see the member IDs that you automatically import. You can all use this time to define you Hierarchies.	

At this step, you connect external view to create a live actual data foundation

3.7 Create the live forecast reference view

Step	Action	Screenshot
-	A seamless planning model can integrate external data as read-only live versions. Planning versions continue to be created using data action. But, for the purposes of this proof of concept, we are also going to create a Forecast version based on drivers and calculations executed live in SAP Datasphere.	
3.7.1	Define a Public dimension to persist your Planning Drivers. Navigate to Modeler, Choose Public Dimensions Table Select Create dimensions Table Choose Start with an empty Model Select Next	
3.7.2	Choose SAP Datasphere, as your data storage location, and click on Create This will also make dimension data immediately available for further “data builder” developments.	

3.7.3	Add an attribute “DRIVER” to your new dimension.	▼ Properties Generic Forecast Driver -> Edit Property ID: * DRIVER Description: Forecast Driver Type: Decimal Decimal Places: 4 ☒ Value is mandatory OK Cancel																								
3.7.4	Expose dimensions Table as dimensions and Text Table in SAP Datasphere	Dimension Table ✕ ▼ Data Storage (SAP Datasphere) SAP Datasphere Space: USR0570801 Exposure in SAP Datasphere ⓘ Status: EXPOSED Business Name: SP_SEAMLESSPLAN_SHOPIFY_D001 Technical Name: SP_SEAMLESSPLAN_SHOPIFY_D001 Edit...																								
3.7.5	Define 4 Drivers for your forecast planning. You can keep drivers = 0	Dimension Table SP_SEAMLESSPLAN_SHOPIFY_D001 <table><thead><tr><th></th><th>Member ID *</th><th>Description</th><th>Forecast Driver *</th></tr></thead><tbody><tr><td>1</td><td>#</td><td>Unassigned</td><td>0</td></tr><tr><td>2</td><td>D001</td><td>Gross Sales Variance %</td><td>0</td></tr><tr><td>3</td><td>D002</td><td>Discount Variance %</td><td>0</td></tr><tr><td>4</td><td>D003</td><td>Active Customers Variance %</td><td>0</td></tr><tr><td>5</td><td>D004</td><td>New Customers Variance %</td><td>0</td></tr></tbody></table>		Member ID *	Description	Forecast Driver *	1	#	Unassigned	0	2	D001	Gross Sales Variance %	0	3	D002	Discount Variance %	0	4	D003	Active Customers Variance %	0	5	D004	New Customers Variance %	0
	Member ID *	Description	Forecast Driver *																							
1	#	Unassigned	0																							
2	D001	Gross Sales Variance %	0																							
3	D002	Discount Variance %	0																							
4	D003	Active Customers Variance %	0																							
5	D004	New Customers Variance %	0																							
-	Since you choose to expose your public dimension in SAP Datasphere you are capable to see this data in “Data Builder”																									
3.7.6	Find Public dimension tables and move it to ‘5 – SAC Planning’ folder.	List Objects In: AI > C3Shopify Seamless Planning > C3 - SAC Planning (7) <table><thead><tr><th>Business Name</th><th>Technical Name</th><th>Type (Semantic Usage)</th></tr></thead><tbody><tr><td>SP_SEAMLESSPLAN_SHOPIFY_D001_TEXT</td><td>sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001_TEXT</td><td>Local Table (Text)</td></tr><tr><td>SP_SEAMLESSPLAN_SHOPIFY_D001</td><td>sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001</td><td>Local Table (Dimension)</td></tr></tbody></table>	Business Name	Technical Name	Type (Semantic Usage)	SP_SEAMLESSPLAN_SHOPIFY_D001_TEXT	sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001_TEXT	Local Table (Text)	SP_SEAMLESSPLAN_SHOPIFY_D001	sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001	Local Table (Dimension)															
Business Name	Technical Name	Type (Semantic Usage)																								
SP_SEAMLESSPLAN_SHOPIFY_D001_TEXT	sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001_TEXT	Local Table (Text)																								
SP_SEAMLESSPLAN_SHOPIFY_D001	sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001	Local Table (Dimension)																								

3.7.7	You can create a new graphical view in “3 – Propagation” folder to join Drivers properties to Driver Texts Technical ID: 3VR_SP_DRIVERS Description: Seamless Planning Drivers Note: this is optional for reporting.	 <table border="1"> <thead> <tr> <th>Data: Seamless Plan... (3)</th> <th>Errors (0)</th> <th>Data Validation</th> </tr> </thead> <tbody> <tr> <td>#</td> <td>DESCRIPTION</td> <td>*LANG DRIVER</td> </tr> <tr> <td>D001</td> <td>Gross Sales Variance %</td> <td>EN 0.0000</td> </tr> <tr> <td>D002</td> <td>Discount Variance %</td> <td>EN 0.0000</td> </tr> <tr> <td>D003</td> <td>Active Customers Variance %</td> <td>EN 0.0000</td> </tr> <tr> <td>D004</td> <td>New Customers Variance %</td> <td>EN 0.0000</td> </tr> </tbody> </table>	Data: Seamless Plan... (3)	Errors (0)	Data Validation	#	DESCRIPTION	*LANG DRIVER	D001	Gross Sales Variance %	EN 0.0000	D002	Discount Variance %	EN 0.0000	D003	Active Customers Variance %	EN 0.0000	D004	New Customers Variance %	EN 0.0000
Data: Seamless Plan... (3)	Errors (0)	Data Validation																		
#	DESCRIPTION	*LANG DRIVER																		
D001	Gross Sales Variance %	EN 0.0000																		
D002	Discount Variance %	EN 0.0000																		
D003	Active Customers Variance %	EN 0.0000																		
D004	New Customers Variance %	EN 0.0000																		
3.7.8	Go to folder “2 – Harmonisation”. Create new SQL view “2VRShopify_SAC_Planning_Forecast” This view will provide forecast for the following accounts: -- A001 Gross sales with Tax -- A002 Discounts with tax -- A003 Taxes -- A004 Cost of Goods Sold -- S001 Active Customers -- S002 New Customers You can see similarities to “2VRShopify_SAC_Planning_Actual” view, but this time we multiply actuals by drivers variances.	<pre>-- A001 Gross sales with Tax SELECT TO_NVARCHAR(ADD_MONTHS(L."Calendar_Month", 3), 'YYYYMM') AS "Calendar_Month", L."Store", L."CUSTOMER_GROUP", L."MERCH_CATEG", 'A001' AS "ACCOUNT_ID", CAST(CAST(L."GROSS_LINE_AMOUNT" AS DECIMAL(15,4)) * (1 + CAST(D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" AS DECIMAL(15,4)) / 100) AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" L INNER JOIN "sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001" D ON D."SP_SEAMLESSPLAN_SHOPIFY_D001_ID" = 'D001' UNION ALL -- A002 Discounts with tax SELECT TO_NVARCHAR(ADD_MONTHS(L."Calendar_Month", 3), 'YYYYMM') AS "Calendar_Month", L."Store", L."CUSTOMER_GROUP", L."MERCH_CATEG", 'A002' AS "ACCOUNT_ID", CAST(CAST(L."DISCOUNT_LINE_ITEM" AS DECIMAL(15,4)) * (1 + CAST(D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" AS DECIMAL(15,4)) / 100) AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" L INNER JOIN "sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001" D ON D."SP_SEAMLESSPLAN_SHOPIFY_D001_ID" = 'D002' UNION ALL -- A003 Taxes SELECT TO_NVARCHAR(ADD_MONTHS(L."Calendar_Month", 3), 'YYYYMM') AS "Calendar_Month", L."Store", L."CUSTOMER_GROUP", L."MERCH_CATEG", 'A003' AS "ACCOUNT_ID", CAST(CAST(L."TAX_LINE_ITEM" AS DECIMAL(15,4)) * (1 + CAST(D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" AS DECIMAL(15,4)) / 100) AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" L INNER JOIN "sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001" D ON D."SP_SEAMLESSPLAN_SHOPIFY_D001_ID" = 'D001' UNION ALL -- A004 Cost of Goods Sold SELECT TO_NVARCHAR(ADD_MONTHS(L."Calendar_Month", 3), 'YYYYMM') AS "Calendar_Month", L."Store", L."CUSTOMER_GROUP", L."MERCH_CATEG", 'A004' AS "ACCOUNT_ID", CAST(CAST(L."COGS" AS DECIMAL(15,4)) * (1 + CAST(D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" AS DECIMAL(15,4)) / 100)</pre>																		

		<pre> AS DECIMAL(15,4)) AS "AMOUNT", CAST(0 AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Line_Items" L INNER JOIN "sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001" D ON D."SP_SEAMLESSPLAN_SHOPIFY_D001_ID" = 'D001' UNION ALL -- S001 Active Customers SELECT TO_NVARCHAR(ADD_MONTHS(L."Calendar_Month", 3), 'YYYYMM') AS "Calendar_Month", L."Store", '#', '#', 'S001' AS "ACCOUNT_ID", CAST(0 AS DECIMAL(15,4)) AS "AMOUNT", CAST(CAST(COUNT(DISTINCT L."CUSTOMER_ID") AS DECIMAL(15,4)) * (1 + CAST("SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" AS DECIMAL(15,4)) / 100) AS DECIMAL(15,4)) AS "INDEX" FROM "2VRShopify_Order_Header" L INNER JOIN "sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001" D ON D."SP_SEAMLESSPLAN_SHOPIFY_D001_ID" = 'D003' GROUP BY L."Calendar_Month", L."Store", D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" UNION ALL -- S002 New Customers SELECT TO_NVARCHAR(ADD_MONTHS(T."Calendar_Month", 3), 'YYYYMM') AS "Calendar_Month", T."Store", '#', '#', 'S002', CAST(0 AS DECIMAL(15,4)) AS "AMOUNT", CAST(CAST(SUM(T.NEW_CUSTOMER) AS DECIMAL(15,4)) * (1 + CAST(D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" AS DECIMAL(15,4)) / 100) AS DECIMAL(15,4)) AS "INDEX" FROM (SELECT "Calendar_Month", "Store", CASE WHEN LAG("ORDER_NUMBER") OVER (PARTITION BY "CUSTOMER_ID" ORDER BY "SHOPIFY_CREATED_AT", "ORDER_NUMBER") IS NULL THEN 1 ELSE 0 END AS NEW_CUSTOMER FROM "2VRShopify_Order_Header") T INNER JOIN "sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001" D ON D."SP_SEAMLESSPLAN_SHOPIFY_D001_ID" = 'D004' GROUP BY T."Calendar_Month", T."Store", D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" </pre>
--	--	--

3.7.9

Finally define the metadata of your graphical view.

- fact table
- two measures
- five attributes

This is Planning model structure

Model Properties

Shopify Seamless Planning (SAC...

Business Name:
Shopify Seamless Planning (SAC Forecast)

Technical Name:
2VRShopify_SAC_Planning_Forecast

Language:
SQL (Standard Query)

Release State:
Not Released

Semantic Usage:
Fact

Expose for Consumption
ON

Run in Analytical Mode:
OFF

Status:
Deployed

Attributes (5)

Tt Calendar Month

Tt Store

Tt CUSTOMER_GROUP

Tt MERCH_CATEG

Tt ACCOUNT_ID

Measures (2)

AMOUNT SUM

INDEX SUM

R0570801 | View - Shopify Seamless Planning (SAC Forecast) * X

EditTools

41L."CUSTOMER_GROUP",

42L."MERCH_CATEG",

43"AB03" AS "ACCOUNT_ID",

44CAST(

45CAST(L."TAX_LINE_ITEM" AS DECIMAL(15,4)) *

46(1 + CAST(D."SP_SEAMLESSPLAN_SHOPIFY_D001_DRIVER" AS DECIMAL(15,4)) / 100)

47AS DECIMAL(15,4)) AS "AMOUNT",

48CASTER AS DECIMAL(15,4)) AS "INDEX"

49FROM "2VRShopify_Line_Items" L

50INNER JOIN "sap.sac.SP_SEAMLESSPLAN_SHOPIFY_D001" D

Data: Shopify Seaml... (38)Errors (0)Data Validation

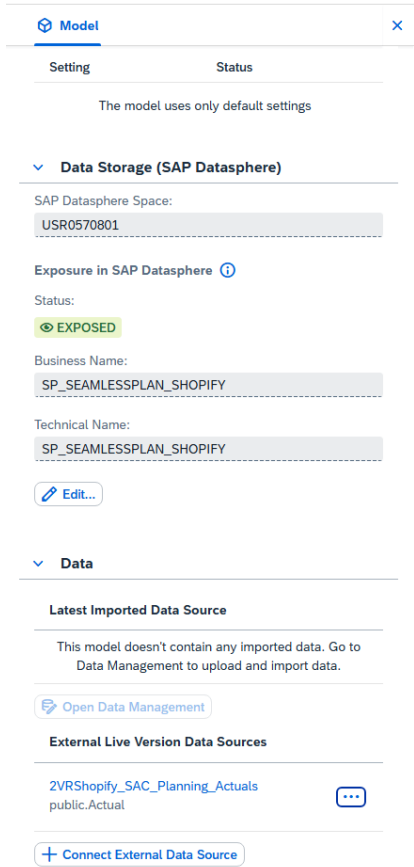
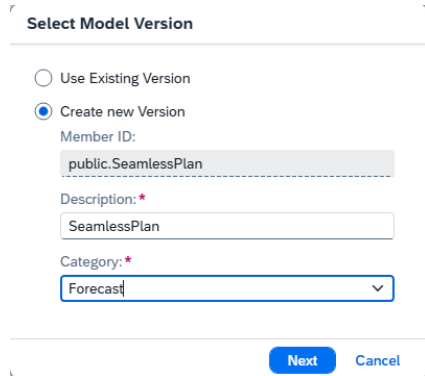
Calendar Month	Store	CUSTOMER_GROUP	MERCH_CATEG	ACCOUNT_ID	AMOUNT	INDEX
202612	0001	Portugal	Pan	A001	99.9000	0.0000
202612	0001	Portugal	Set	A001	62.0500	0.0000
202612	0001	Netherlands	Board	A001	20.1500	0.0000
202612	0001	Netherlands	Pan	A001	99.9000	0.0000
202612	0001	Portugal	Maker	A001	92.6500	0.0000
202612	0001	Portugal	L	A001	12.9000	0.0000
202612	0001	Portugal	Kettle	A001	63.6500	0.0000

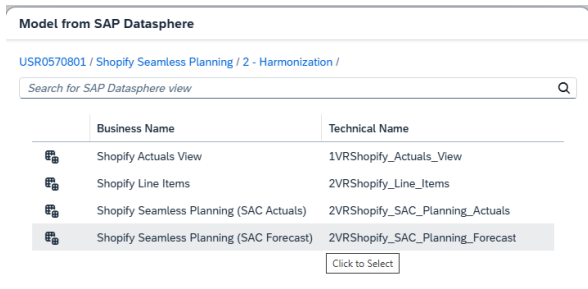
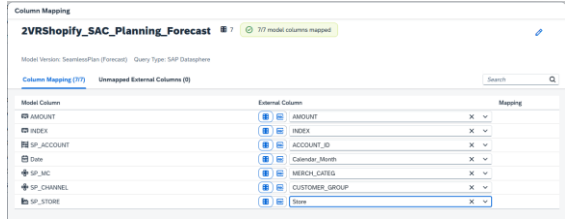
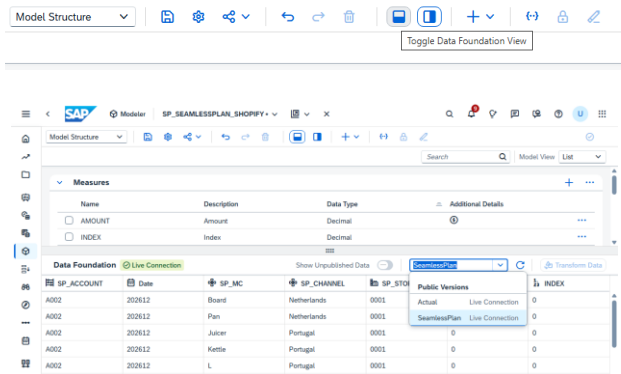
At this step, you create proper sql view to deliver live forecast data to planning model.

3.8 Connect the live forecast reference version

Step	Action	Screenshot
------	--------	------------

Pedro Filipe • Proof of Concept

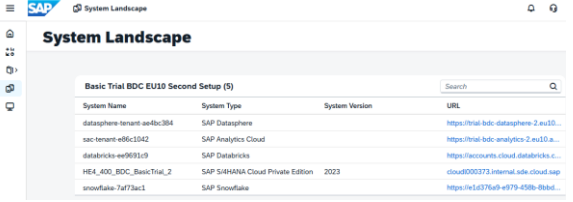
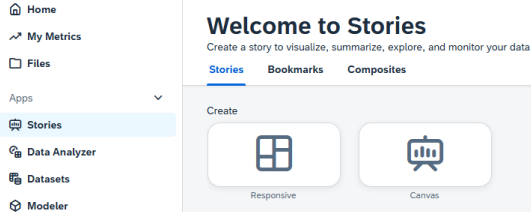
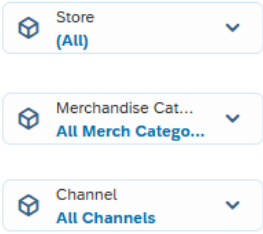
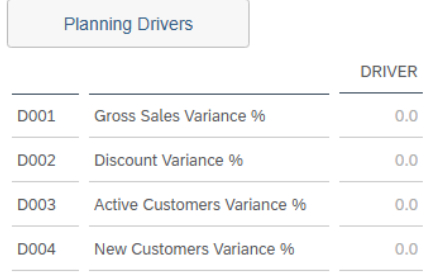
-	Now we want to extend our Seamless Planning advantages, to have a new read-only live forecast reference version. Under right side details panel, choose Model Settings	
3.8.1	We are going to connect an external live version datasource to a new Forecast Version. Select Connect External Data Source	
3.8.2	Create New “SeamlessPlan” Next	

3.8.3	Choose your SAP Datasphere View to integrate a Live Forecast version in your planning model. Remember this view already multiplies actuals by driver variances.	
3.8.4	Map SAP Datasphere fields to Planning model Dimensions and Measures. Select OK	
3.8.5	“Toggle Data Foundation View” to display Live Forecast data.	

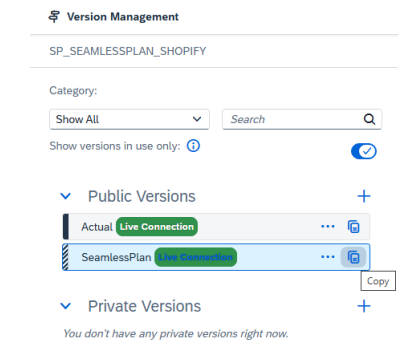
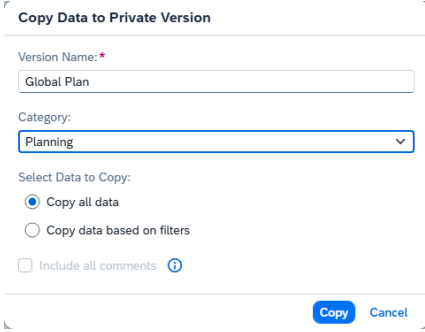
At this step, you connect external view to create a live forecast data foundation

3.9 Create the SAP Analytics Cloud planning story

Step	Action	Screenshot
-	SAP Analytics Cloud data consumption	
3.9.1	The final step is to create the Planning Story. For sure this is the most visible part, but the core modelling work has already been	

	completed in SAP Datasphere and Planning model.																
3.9.2	Under BDC, open System Landscape. You get the access URLs to access SAP Datasphere (data modelling) and SAC (planning and reporting). Select the SAP Analytics Cloud url.																
3.9.3	Navigate to Stories, Select Create New Story Choose Responsive, Canvas or any of existing Templates Select Next																
-	Define your story with 4 distinct areas																
3.9.4	Area 1: Global Filters 3 main planning levels: - Stores - Merchandise Categories - Customer Groups / Channels																
3.9.5	Area 2: Planning Drivers This is where you can change your drivers by changing them directly in public dimension To maintain master data with scripting follow SAP learning guidelines	 <table border="1"> <thead> <tr> <th></th><th></th><th>DRIVER</th></tr> </thead> <tbody> <tr> <td>D001</td><td>Gross Sales Variance %</td><td>0.0</td></tr> <tr> <td>D002</td><td>Discount Variance %</td><td>0.0</td></tr> <tr> <td>D003</td><td>Active Customers Variance %</td><td>0.0</td></tr> <tr> <td>D004</td><td>New Customers Variance %</td><td>0.0</td></tr> </tbody> </table>			DRIVER	D001	Gross Sales Variance %	0.0	D002	Discount Variance %	0.0	D003	Active Customers Variance %	0.0	D004	New Customers Variance %	0.0
		DRIVER															
D001	Gross Sales Variance %	0.0															
D002	Discount Variance %	0.0															
D003	Active Customers Variance %	0.0															
D004	New Customers Variance %	0.0															

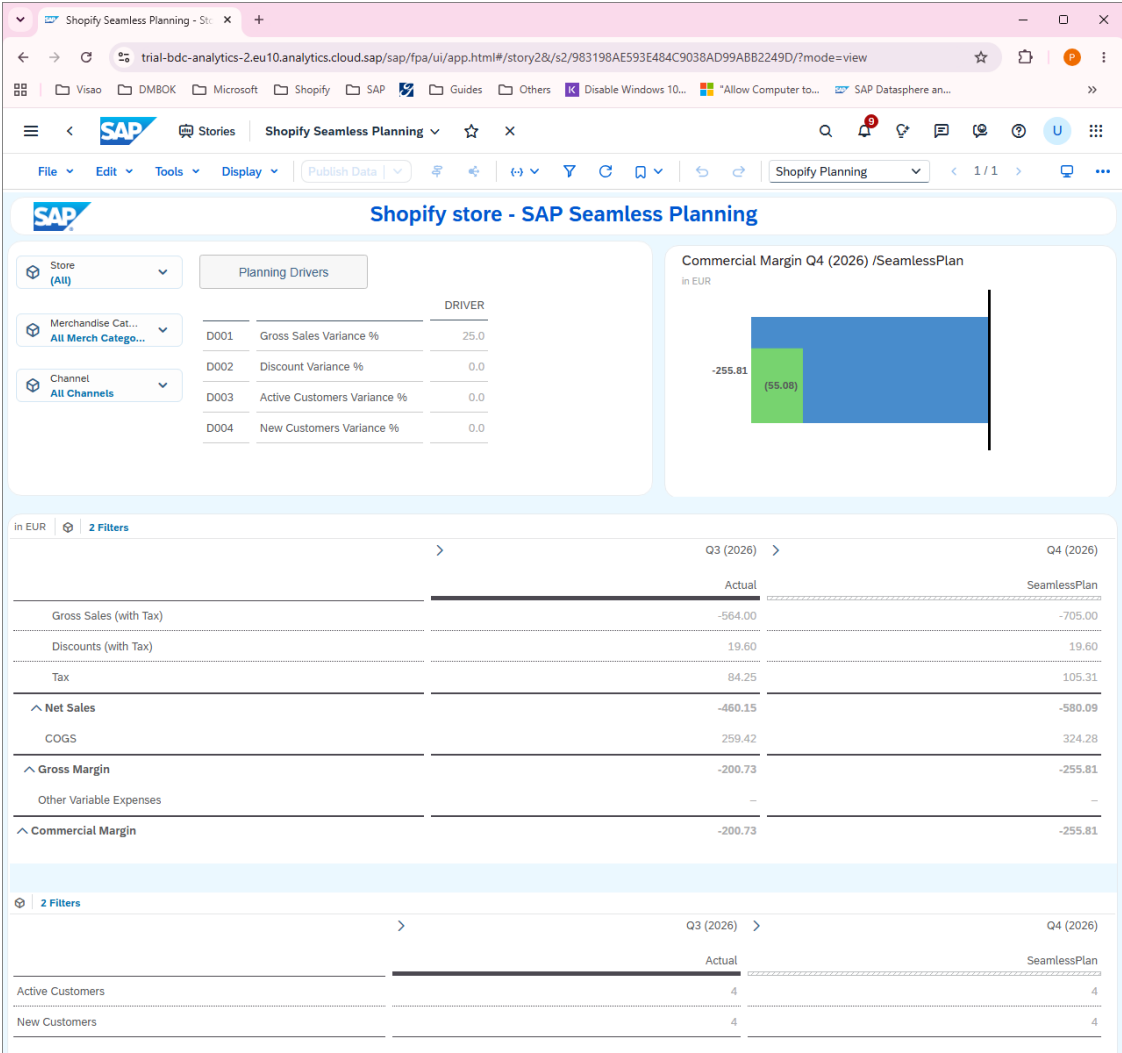
3.9.6	Area 3: Chart (Overview) Compare the current quarter Live Actuals with the next quarter Live Forecast	Commercial Margin Q4 (2026) /SeamlessPlan in EUR </
-------	---	--

	Finally, you need to enter the costs of a new marketing campaign and start planning at different levels (example Merchandise Category Managers) Copy your Live Forecast into a new version, ready for manual planning.	
3.9.10	Copy the data and use the story to enter new costs in table.	

At this step, the end-to-end proof of concept is complete.

Execution flow:

- ➔ Actual Shopify data flows from Shopify stores to SAP BDC Data product
- ➔ SAP Seamless Planning model, allows live actual sales to be integrated in planning process.
- ➔ SAP Datasphere Views allow forecast values to be recalculated Forecast values on the fly



4. Lessons learned and conclusions

4.1 Start with the target planning grain

The most important design decision is the grain shared by actuals, reference forecasts and writable plans. Date, store, customer group, merchandise category, account and measure semantics must be aligned before live-version mapping.

4.2 Model compatibility is more important than connector speed

The Shopify source is key-figure based, whereas the SAC solution uses an Account dimension. Converting source measures into an ACCOUNT_ID and value structure was therefore essential. The UNION ALL approach is explicit, easy to test and suitable when UNPIVOT syntax is unavailable in the SQL View editor.

4.3 Separate live reference data from writable planning data

External live versions are best treated as read-only reference foundations. In this solution, Actuals and the calculated forecast baseline are supplied by SAP Datasphere. The forecast baseline is then copied to a writable SAC version when planners need manual adjustments, allocations or collaboration.

4.4 Use SAP Datasphere for reusable data engineering

Calculations that are shared across planning and analytics, such as line-level tax allocation, discount allocation, COGS enrichment and customer classification, belong in a governed SAP Datasphere layer. SAC remains responsible for the planning model, account logic, versions, data entry and the planning experience.

4.5 Seamless does not mean model-free

Seamless planning reduces data movement, but it does not remove the need for semantic governance. Master-data IDs, hierarchies, Account Formulas, etc, still require explicit design and reconciliation.

4.6 Public dimensions can act as lightweight driver stores

Persisting forecast drivers in a public dimension makes the assumptions available in both SAC and SAP Datasphere. This is effective for a compact proof of concept. For production driver maintenance options may be considered.

4.7 Validate performance and security end to end

A production solution should test query response times at realistic volumes and verify authorisations in both SAC and SAP Datasphere. Live integration removes scheduled imports, but source freshness, lineage, reconciliation and operational monitoring remain necessary.

4.8 Conclusion

The proof of concept shows that SAC Seamless Planning can combine governed SAP Datasphere actuals with the SAC planning experience without replicating actual data into the planning model. It also demonstrates how SAP Datasphere can calculate a live forecast baseline from centrally maintained drivers. The strongest

production pattern is hybrid: use live versions for trusted reference data, use writable SAC versions for human planning, and place reusable transformation logic in SAP Datasphere. This keeps responsibilities clear while reducing unnecessary data movement and maintenance.

4.9 Recommended next steps

Before production deployment, replace demo data with governed sources, formalise naming and transport conventions, implement role-based access, define reconciliation controls, performance-test the live views, and document which calculations belong in SAP Datasphere versus SAC.

5. References and publication notes

- SAP Help Portal, About Seamless Planning.
- SAP Help Portal, Understand SAP Analytics Cloud Planning Data Storage Configuration in SAP Datasphere.
- SAP Help Portal, About Model Configurations.
- SAP Trial environment templates.
- All screenshots were captured in the SAP Trial environments. Review screenshots for tenant names, user IDs, URLs and other information before publication.