Integration Guide | PUBLIC
Document Version: 1.1.1 – 2026-05-03

Integrating Sales Order History Data from SAP S/4HANA Cloud Private Edition or SAP S/4HANA or SAP ERP to SAP Integrated Business Planning

Integrating SAP IBP with SAP S/4HANA Cloud Private Edition and SAP S/4HANA and SAP ERP Using SAP Cloud Integration

Content

1 Document History

The following table provides an overview of the most important changes.

Version	Date	Description
1.1.1	May 3, 2026	Updated the dafault value for the maximum package size.
1.1.0	November 26, 2025	Updated document with minor refinements.
1.0.0	August 1, 2025	Initial version

2 Introduction

Using the *Integrate Sales Order History Data from Add-on to SAP IBP* integration flow, you can integrate data from SAP S/4HANA Cloud Private Edition or SAP S/4HANA or SAP ERP (source system) to SAP Integrated Business Planning (SAP IBP).

The *Integrate Sales Order History Data from Add-on to SAP IBP* integration flow collects sales order history data from the source system using the SAP S/4HANA, Supply Chain Integration Add-On for SAP Integrated Business Planning or the SAP ERP Supply Chain Integration Add-On for SAP Integrated Business Planning (add-on) and transfers it into an existing planning area in SAP IBP. Using this integration flow, you can make sales order history data available for running demand forecasting in SAP IBP.

If you want to call the *Integrate Sales Order History Data from Add-on to SAP IBP* integration flow with the *Process Direct* connection type, use the **Integrate_Sales_Order_History_Data_from_Add-on_to_SAP_IBP** address.

3 Prerequisites

- You have already set up the connection and did the necessary steps to properly set up the environment: the add-on, the SAP Cloud Connector, SAP Business Technology Platform (SAP BTP), the SAP Integration Suite, SAP IBP, and the source system. You can find an end-to-end connection setup guide in SAP Note 3628813.
- You have already integrated the necessary master data to SAP IBP.

Related Information

SAP S/4HANA, supply chain integration add-on for SAP Integrated Business Planning
SAP ERP, supply chain integration add-on for SAP Integrated Business Planning
Cloud Connector

4 Configuring the Integration Flow

To be able to transfer data between SAP IBP and the source system, you first need to set up and configure the connection between these systems.

4.1 Configuring the Authentication

The integration flow connects the add-on and the SAP IBP system. Connections, including the authentication method, must be created and configured differently depending on the respective system and the selected authentication method.

Authentication Methods for Connecting to SAP IBP

You can choose basic authentication or the propagation of a technical user or setting up principal propagation when connecting to SAP IBP. You can configure the authentication method during the configuration of the destination. You can set the name of the destination using the `Destination for SAP IBP` parameter of the integration flow. For more information, see Setting Up the Integration with SAP S/4HANA Cloud Private Edition or SAP S/4HANA or SAP ERP.

Authentication Methods for Connecting to the Add-On

The following authentication methods are available when connecting to the source system:

- Basic authentication
- Technical user propagation
- Principal propagation

To set up the authentication, you need to follow these steps:

1. Connect your SAP BTP subaccount in the SAP Cloud Connector.
2. Configure `Cloud-To-On Premises` mapping in the SAP Cloud Connector.
3. Enable resources for the mapped system.
4. Create RFC destination in SAP BTP.

For more information, see Cloud Connector

4.2 Data Mapping

You can map the fields of the `/IBP/ACTUALS_QTY_CI_KF` Extractor to attributes in SAP IBP for data integration.

The key externalized parameters of the integration flow for default mapping are the following:

- *Source Fields from Add-On*
  It must contain the requested fields from the *Extractor*, separated by a comma.
- *Target Fields in SAP IBP*
  It must contain the target fields in SAP IBP, separated by a comma.
- *Field Mapping*
  It must contain the field mappings between the source fields and the target fields, in an `SOURCE:TARGET` format. For example, `PRDID:I_PRDID,LOCID:I_LOCID,ACTUALSQUANTITY:I_ACTUALSQTY,'DUMMY':I_TSPCUSTID`
  You can map constant values in the mapping between single quotes. For example: `'DUMMY':I_TSPCUSTID`. You can't have a quote in the constant value.

The following default data mapping is available in the integration flow:

Field in Extractor	Field in SAP IBP	Further Hints
`PRDID`	`I_PRDID`	
`LOCID`	`I_LOCID`	
`SDCUSTID`	`I_TSPCUSTID`	
`KEYFIGUREDATE`	`KEYFIGUREDATE`	
`ACTUALSQTY`	`I_ACTUALSQTY`	The value is considered in the base unit of measure.

4.3 Defining Additional Parameters

You can configure your integration flow by setting externalized parameters. You can either create your own wrapper integration flow to alter these settings or directly change the values in the SAP-delivered integration flow. In the integration flow editor, click *Configure*, then *More* and assign values for the parameter names to set up your integration flow. If you call the integration flow using the *Process Direct*, use the parameter ID for the *Connection Address*.

The following list contains the delivered externalized parameters and their default values:

Parameter Name	Parameter ID	Default Value	How to configure the parameter?
`Batch Command`	`BatchCommand`	`INSERT_UPDATE`	Define the SAP IBP batch processing command. Accepted values: `INSERT_UPDATE`, `REPLACE`, `DELETE`.
`Batch Name`	`BatchName`	`AddOnActualsQuantity ${header.SAP_MplCorrelationId}`	Define the name of the data batch. This name also identifies the corresponding job in the *Data Integration Jobs* app. Mandatory parameter.
`Custom Mapping Extension Integration Flow Address`	`CustomMappingExtension IntegrationFlowAddress`		Using this parameter you can include a custom, extended mapping to the integration flow.
`Date From`	`DateFrom`	`xsd:yearMonthDuration('-P2Y') + xsd:date(substring($IFlowStartTimestamp,1,10))`	Define the starting date of the data to be integrated from the Add-on. The default starting date is two years before the current date. You can set a constant value in single or double quotes in a **`yyyymmdd`** format. **Example** `'20160101'` You can also use XPath expressions which use the date format **`"yyyy-mm-dd"`**, when configuring the integration flow. If you don't want to filter for KEYFIGUREDATE, you have to set an empty value between quotes.

Parameter Name	Parameter ID	Default Value	How to configure the parameter?
`Date To`	`DateTo`	`xsd:date(substring($IFlowStartTimestamp,1,10))`	Define the last date of the data to be integrated from the add-on. The default last date is the current date. You can set a constant value in single or double quotes in a **`yyyymmdd`** format. **Example** `'20160101'` You can also use XPath expressions which use the date format **`"yyyy-mm-dd"`**, when configuring the integration flow. If you don't want to filter for KEYFIGUREDATE, you have to set an empty value between quotes.
`Destination for Add-on`	`DestinationforAddon`		Enter the SAP BTP *Destination Name* which points to the source system. Mandatory parameter.
`Destination for SAP IBP`	`DestinationforSAPIBP`		Enter the SAP BTP *Destination Name* which points to the target SAP IBP system. Mandatory parameter.
`Field Mapping`	`FieldMapping`	`PRDID:I_PRDID,LOCID:I_LOCID,ACTUALSQUANTITY:I_ACTUALSQTY,SDCUSTID:I_TSPCUSTID`	Define the mapping of the source and target fields.
`File Name`	`FileName`		Define a file name. This name also identifies the corresponding file in the *Data Integration Jobs* app.

Parameter Name	Parameter ID	Default Value	How to configure the parameter?
Filter for the Data Source	FilterForTheDataSource	SDDELIVERYSTATUS EQ 'C', SKIP_PD_SELECT EQ 'X'	You can define filters for your data extraction. Set SKIP_PD_SELECT to **'X'** to make sure purchase orders are **not** read. Set SDDELIVERYSTATUS to **'C'** to make sure that only completely processed sales orders are read.
MRP Type Filters	MRPTypeFilters	X0	You can filter for MRP type.
Parallel Data Transfer	ParallelDataTransfer	true	
Planning Area	PlanningArea	I_SAPIBP2	Define the planning area in SAP IBP you want to integrate data into. Mandatory parameter.
Planning Area Version	PlanningAreaVersion		Define the version of the target planning area in SAP IBP.
Post-Fetch Extension Integration Flow Address	PostFetchFilterExtensionAddress		You can define a custom filter expression.
Read Package Size in Mb	ReadPackageSizeinMb	10	You can specify the maximum package size coming from the add-on. Valid values are integers greater than zero. For more information about package support, see SAP Note 2196500
Source Fields from Add-on	SourceFieldsfromAddon	PRDID,LOCID,SDCUSTID,KEYFIGUREDATE,ACTUALSQUANTITY	Define the sales order attributes that you want to retrieve from the extractor.
Source Time Profile ID from Add-on	SourceTimeProfilefromAddon		Define the source time profile ID in the add-on. Valid values are integers. Mandatory parameter.

Parameter Name	Parameter ID	Default Value	How to configure the parameter?
`Source Time Profile Level from Add-on`	`SourceTimeProfileLevelfromAddon`	2	Define the time aggregation level to be used in the add-on. We recommend keeping the source and target time profile levels in synch. Valid values are integers. Mandatory parameter.
`Target Fields in SAP IBP`	`TargetFieldsInSAPIBP`	`I_PRDID,I_LOCID,I_TSPCUSTID,KEYFIGUREDATE,I_ACTUALSQTY`	Define the target fields to which you want to upload data.
`Target Time Profile Level in SAP IBP`	`TargetTimeProfileLevelinSAPIBP`	2	Define the target time profile level in SAP IBP for disaggregation.

4.4 Time Aggregation

You can configure a `Time Profile Level in SAP IBP` and upload the SAP IBP `Time Profile` in the Add-on. By synchronizing the time profile levels, you can process data without aggregation attempts.

You have the option to configure parameters such as the `Target Time Profile Level in SAP IBP` and the `Source Time Profile ID` and `Source Time Profile Level` in the source system. These parameters can be combined in different ways.

SAP IBP only offers to integrate key figure data on the base time profile level or can perform disaggregation when data are uploaded on a higher time profile level. Loading data on a time profile level that's lower than the base time profile level of the key figure results in failure. To avoid this, we recommend that you process data without aggregation. This can be achieved by setting the combination through matching the time level value pairs.

Example

If the following conditions are fulfilled, the data will be disaggregated to the key figure's base time profile level:

- The key figure configuration allows for disaggregation.
- The time profile level **`4`** means **`month`** in SAP IBP.
- The `Target Time Profile Level in SAP IBP` is set to **`4`**.
  This indicates that you want to load the data on a monthly level.
- The base time profile level of the loaded key figure is technical week.
- The `Source Time Profile Level in Add-on` is set to **`4`** (monthly level).

4.5 Filtering Configuration

Use a simplified filter to narrow down the data load you want to integrate. You can filter for individual entries or ranges of entries, or entries in a specific pattern. For more complex filters, use the *Post Fetch Extension*, where you can define custom filtering in a groovy script file.

As the filter conditions are translated into ABAP selection tables in the background, there are a few rules and limitations that must be followed.

Using the Filter for the Data Source parameter for individual entries, you can add the field and the exact values you want the filter for. The filtered values must always be put between a pair of single quotation marks, and each filter condition must be separated by a comma.

The following operators are supported by the integration flow: EQ, NE, CP, NP, BT, NT.

If you filter for **`ExampleProduct1`**, only data that contains ExampleProduct1 is integrated.

> ### Example
>
> *Filter for the Data Source:* **`ExampleProduct1`**

You can filter for different values for different fields.

> ### Example
>
> *Filter for the Data Source:* **`PRDID EQ 'ExampleProduct1', LOCID EQ 'ExampleLocation1'`**

If you want to filter for multiple values of the same field, the values should be written into simple conditions or you can specify them as a range.

> ### Example
>
> *Filter for the Data Source:* **`PRDID BT 'ExampleProduct1' 'ExampleProduct3'`**
>
> If you want to filter for multiple values of multiple fields, similarly to the example above, the conditions should be written one after another.

You can also filter records based on the condition that a field contains a specific substring.

> ### Example
>
> *Filter for the Data Source:* **`PRDID CP 'EXAMPLE_PHONE_*'`**

This will filter for every product that starts with 'EXAMPLE_PHONE_'. The **`*`** sign can only come to the end of the string.

> ### Note
>
> The order of the simple conditions does not matter, as they will be automatically grouped in the selection table.
>
> In the background, in the ABAP selection table, two rows are related in the following way:
>
> - Selections to the same field will have an OR relation

- Selections to different fields will have an AND relation

Because of this logic, using the operators AND and OR are not allowed.

In addition to these filtering techniques, there are also specific methods available for working with date fields in the system.

Filtering for KEYFIGUREDATE can be done in three ways:

- Set an XPath value for the *Date From* and *Date To* parameters.
- Set a constant value for the *Data From* and *Date To* parameters in a **`yyyymmdd`** or a **`yyyy-mm-dd`** format.
- Set the *Filter for the Data Source* to a value where it is filtering for the KEYFIGUREDATE in a **`yyyymmdd`** format.

> ### Example
>
> Filter for the Data Source: KEYFIGUREDATE BT '20160101' '20250101'

If you don't want to filter for KEYFIGUREDATE, you must set the *Date From* and *Date To* parameter to an empty value between single or double quotes.

If you are using the *Date From* and *Date To* parameters, the filtering only happens when both parameters are filled.

4.6 Working with Extensions

Parameters for extensibility allow you to specify additional attribute mappings and filters that can be used to integrate data from external sources. You can further modify the way data is mapped to integrate data by using field extensions.

In general, the extensions are custom defined integration flows, that will be called by a *Process Direct* call. On the *Configuration* tab of the integration flow, in the *Post-Fetch Extension Integration Flow Address* and/or the *Custom Mapping Extension Integration Flow Address* fields, you must only add the custom extension integration flow ID.

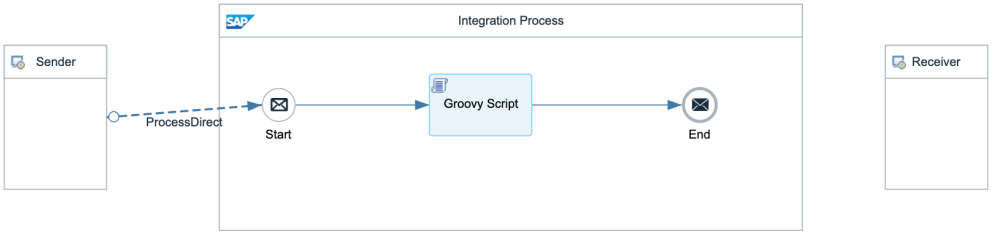
4.6.1 Post Fetch Extension

The *Post Fetch Extension* is primarily designed for specifying additional, more complex filter expressions via Groovy script, but generally can be used for any kind of custom extensibility implementation to filter or alternate the extracted data.

The SAP Integration Suite offers high-level flexibility to implement custom logic with various tools.

> ### Example
>
> The following example shows you how to manipulate data via a Groovy script.

A simple integration flow has to be defined, which is called by a *Process Direct* call. On the Connection tab, in the *Address* field, you must enter the ID of your extension integration flow.



In the integration process, you need to define a Groovy script.

Example

In our example, we are filtering for `ExampleProduct1` and modifying the corresponding `Actuals Quantity` by adding **5** to it.

```
import com.sap.gateway.ip.core.customdev.util.Message
import groovy.json.JsonSlurper
import groovy.json.JsonBuilder

def Message processData(Message message) {
    // Parse the JSON payload
    def body = message.getBody(java.io.InputStream);
    def jsonPayload = new JsonSlurper().parseText(body.text)

    // Filter the records to include only those where PRDID is 'ExampleProduct' and add 5 to ACTUALSQUANTITY
    jsonPayload.ITEMS = jsonPayload.ITEMS.findAll { item ->
        if (item.PRDID == 'ExampleProduct') {
            item.ACTUALSQUANTITY = (item.ACTUALSQUANTITY as BigDecimal) + 5
            return true
        }
        return false
    }

    // Create a JSON builder to convert the map back to a JSON string
    def jsonOutput = new JsonBuilder(jsonPayload).toPrettyString()

    // Set the modified payload as the new message body
    message.setBody(jsonOutput)

    return message
}
```

Creating the script is either possible in the SAP Integration Suite directly or by uploading a file created in an external editor. When using an external editor, make sure to include the `import com.sap.gateway.ip.core.customdev.util.Message` in the first row.

For more information about the use cases of script, see SAP Cloud Integration - Script Use Cases.

For more information about the SCRIPT APIs, see https://help.sap.com/doc/a56f52e1a58e4e2bac7f7adbf45b2e26/Cloud/en-US/index.html.

4.6.2 Custom Mapping Extension

Similarly to the *Post Fetch Extension*, the *Custom Mapping Extension* can also be implemented as a custom integration flow, called by a *Process Direct* call from the `Integrate Sales Order History Data from Add-on` to SAP IBP integration flow.

> **Note**
>
> The main difference between the *Post Fetch Extension* and the *Custom Mapping Extension* is that the *Post Fetch Extension* keeps the original mapping of the iFlow, while with the *Custom Mapping Extension* you can still modify the data, but you must implement a mapping since the original business integration flow mapping is ignored.

4.7 Scheduling the Integration Flow

You can schedule the execution of the integration flow in *Configure* tab under *Timer*.

There are multiple options available for integration flow scheduling using a *Timer on Cloud Integration* from *Basic* scheduling to an *Advanced* configuration, so that you can choose the option that serves the business needs best. For more information about how to use a *Timer Start Event*, see SAP Cloud Integration - Define a Timer Start Event.

5 Troubleshooting

The following points describe some of the common issues during integration:

- The defined filters are invalid.

  > ✤ Example
  >
  > `PRDID value = "FG226"`

- After choosing *Deploy*, you get an error message: `org.apache.camel.component.directvm.DirectVmConsumerNotAvailableException: No consumers available on endpoint` .
  Reason: Most probably the *Connection Address* is missing in one, or more *Process Direct* calls.
- If there is an error or escalation during the whole integration, you can find the relevant log messages in the substeps filtered by the *Correlation ID* in the *Monitor Messages* app.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap