Integration Guide | PUBLIC
Document Version: 1.5 – 2025-07-11

Integrating Plant Data from SAP S/4HANA Cloud Public Edition to SAP Integrated Business Planning

Integrating SAP IBP with SAP S/4HANA Cloud Public Edition Using SAP Cloud Integration

THE BEST RUN SAP

Content

1 Introduction. 4

2 Prerequisites. 5

3 Configuring the Integration Flow. 6

3.1 Configuring the Authentication. 6

3.2 Data Mapping. 7

3.3 Defining Additional Parameters. 7

3.4 Filtering Configuration. 9

3.5 Working with Extensions. 10

Working with Field Extensions. 11

3.6 Scheduling the Integration Flow. 13

4 Troubleshooting. 14

Document History

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

Version	Date	Description
1.5	July 11, 2025	Updated document subtitle.
1.4	February 3, 2025	Updated document with minor refinements.
1.3	November 25, 2024	The address to call the integration flow by using Process Direct has been added to the Introduction [page 4]. The Filtering Configuration [page 9] chapter has been added.
1.2	June 12, 2024	Added authentication with certificate as an available communication scenario to the Configuring the Authentication [page 6] section. Restructured the Working with Extensions [page 10] chapter.
1.1	May 15, 2024	Minor language and parameter enhancements in Defining Additional Parameters [page 7] section.
1.0	February 16, 2024	Initial version

1 Introduction

Using the integration flow, you can integrate data from SAP S/4HANA Cloud Public Edition to SAP Integrated Business Planning (SAP IBP). Using this data, you can perform demand forecasting in SAP IBP, then integrate the results back to SAP S/4HANA Cloud Public Edition as planned independent requirements.

Data integration between SAP IBP and SAP S/4HANA Cloud Public Edition using the integration flows in the *SAP IBP - Integration with SAP S/4HANA Cloud Public Edition* package is available with SAP IBP 2402 and higher.

Using the *Integrate Plants from SAP S/4HANA Cloud Public Edition to SAP IBP* integration flow, you can integrate plant data from SAP S/4HANA Cloud Public Edition to SAP IBP. Plant data is collected using an OData API. For more information about the API, see the API reference at the SAP Business Accelerator Hub at https://api.sap.com/cdsviews/I_PLANT.

If you want to call the *Integrate Plants from SAP S/4HANA Cloud Public Edition to SAP IBP* integration flow with the Process Direct connection type, use the **Integrate_Plants_from_SAP_S4HANA_Cloud_to_SAP_IBP** address.

2 Prerequisites

We recommend that you have configured frequently used parameters using the *Define Default Values for Data Integration Between SAP IBP and SAP S/4HANA Cloud Public Edition* integration flow.

3 Configuring the Integration Flow

3.1 Configuring the Authentication

The integration flow connects to both the SAP S/4HANA Cloud Public Edition and the SAP IBP system. Connections, including the authentication method, must be created and configured at different places depending on the respective system. Once the connections are created for both directions, you need to configure them in the integration flow under *Configure* > *Receiver*.

Authentication Methods for the Connection to SAP IBP

You can choose basic authentication and authentication with certificate 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.

Authentication Methods for the Connection to SAP S/4HANA Cloud Public Edition

The following authentication methods are available when connecting to SAP S/4HANA Cloud Public Edition:

- Basic authentication
- Client certificate (X.509 certificate)

You can select the authentication method in the integration flow under *Configure* > *Receiver* > *Authentication*. Although there are more options displayed in the list, only basic authentication, authentication with certificate, and client certificate authentication are supported.

The default authentication method is client certificate.

Setting Up the Client Certificate Authentication Method

As a prerequisite of using a client certificate, add and deploy the required key pair to the keystore. You can do so in SAP Integration Suite using the *Keystore* tile in the *Manage Security* section under *Monitoring Artifacts*. For more information, see https://help.sap.com/docs/cloud-integration/sap-cloud-integration/managing-keystore-entries.

If you select authentication using a client certificate when configuring the integration flow, you need to enter the private key alias.

Setting Up the Basic Authentication Method

As a prerequisite of using basic authentication, create and deploy the user credentials type of security material. You can do so in SAP Integration Suite using the *Security Material* tile in the *Manage Security* section under *Monitoring Artifacts*. For more information, see Managing Security Material.

If you select basic authentication when configuring the integration flow, you need to enter the credential name.

3.2 Data Mapping

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

Property in OData Entity	Field in SAP IBP	Further Hints
`Plant`	`LOCID`	
`P`	`LOCTYPE`	The property in the OData entity is fixed as `P`.
`PlantName`	`LOCDESCR`	The property is available from the `Product` entity through the *ProductDescription* > *ProductDescription_Type* > *ProductDescription* navigation.

3.3 Defining Additional Parameters

You can configure your integration flow by setting parameters. In the integration flow editor, click *Configure* > *More* and assign values for the parameter names to set up your integration flow. If you call the integration flow using Process Direct, use the parameter ID.

Parameter Name	Parameter ID	Default Value	How to Configure the Parameter?
`Attributes in SAP IBP`	`AttributesinSAPIBP`	`LOCID,LOCIDDISPLAY, LOCTYPE,LOCDESCR,LOCREGION`	Define the fields of the `Location` master data in SAP IBP into which you want to integrate data.
`Batch Name`	`BatchName`	`Plant: ${header.SAP_MplCorrelationId}`	Define the name of the data batch. This name also identifies the corresponding job in the *Data Integration Jobs* app.
`Destination for SAP IBP`	`DestinationforSAPIBP`	`-keep default-`	Enter the name of the target SAP IBP system into which you want to integrate plant data.

Parameter Name	Parameter ID	Default Value	How to Configure the Parameter?
`Field Extensions`	`FieldExtensions`		Optionally, you can define complex mappings for certain columns. For more information, see Working with Field Extensions [page 11].
`Host for SAP S/4HANA Cloud Public Edition`	`HostforSAPS4HANACloud`	`-keep default-`	Define the base URL of the SAP S/4HANA Cloud API
`Master Data Prefix`	`MasterDataPrefix`	`-keep default-`	Optionally, define a three-character-long prefix to be used in SAP IBP.
`Planning Area`	`PlanningArea`	`-keep default-`	Define the planning area in SAP IBP to which you want data to be integrated.
`Planning Area Version`	`PlanningAreaVersion`	`-keep default-`	Define the version of the target planning area in SAP IBP.
`Plant Filter`	`PlantFilter`	`-keep default-`	Optionally, define filtering for plant data.
`Process Unchanged Data`	`ProcessUnchangedData`	`-keep default-`	If you want to skip unchanged data, regardless of SAP IBP global parameter setting for this behavior, set this parameter to **`false`**. If you want to process unchanged data, regardless of SAP IBP global parameter setting, set this parameter to **`true`**. If you leave this parameter field empty, the SAP IBP global parameter will determine the behavior.

To use the values defined in the *Define Default Values for Data Integration Between SAP IBP and SAP S/4HANA Cloud Public Edition* integration flow, use the `-keep default-` value for the relevant parameters. This is also the default value of all parameters for which you can maintain a reusable default value in the *Define Default Values for Data Integration Between SAP IBP and SAP S/4HANA Cloud Public Edition* integration flow.

3.4 Filtering Configuration

Use a simplified filter to narrow down the data load you want to integrate. You can filter for multiple exact entries or ranges of entries. The simplified filter string is translated to an OData filter string in the background. For more complex filters, use the *Further Filters* parameter that uses a different syntax.

> ### Note
>
> The following examples show the configuration of the externalized filtering parameters. Although we only illustrate *Product Filters*, the same principles apply to the *Plant Filters*, and *Customer Filters* when exchanging product data with the corresponding plant, and customer filter field data.

Using the *Product Filters*, you can filter for exact product data. If you filter for ExampleProduct1, only data that contains ExampleProduct1 is integrated.

> ### Example
>
> *Product Filters:* `ExampleProduct1`
>
> Equivalent OData: Product eq 'ExampleProduct1'

You can use the <–> character to filter for a range of data by separating the start and the end of the range. In this example, you integrate data that contains products between ExampleProduct5 and ExampleProduct9.

> ### Example
>
> *Product Filters:* `ExampleProduct5-ExampleProduct9`
>
> Equivalent OData: Product ge 'ExampleProduct5' and Product le 'ExampleProduct9'

> ### Note
>
> Currently only closed intervals are supported. ExampleEntry5- isn't interpreted as entries that are greater than ExampleEntry5.

If the entry, which isn't a range, contains the <–> character, enclose it in quotation marks.

> ### Example
>
> *Product Filters:* `"Example-Product5"`
>
> Equivalent OData: Product eq 'Example-Product5'
>
> This also applies to Example-Product5 when it is part of the range you want to integrate. This filter includes every entry with a product from Example-Product5 to Example-Product9.
>
> *Product Filters:* `"Example-Product5"-"Example-Product9"`
>
> Equivalent OData: Product ge 'Example-Product5' and Product le 'Example-Product9'

Apply a filter for multiple exact entries or ranges of entries using commas as separators.

> ### Example
>
> In this example, you filter every entry that contains product ExampleProduct1 or any product between ExampleProduct5 and ExampleProduct9.
>
> *Product Filters:* `ExampleProduct1,ExampleProduct5-ExampleProduct9`
>
> Equivalent OData: Product eq 'ExampleProduct1' or Product ge 'ExampleProduct5' and Product le 'ExampleProduct9'

You can use the following filter to include every entry that contains ExampleProduct1, or products between ExampleProduct5 and ExampleProduct9, or Example-Product1, or entries between Example-Product5 and Example-Product9.

> ### Example
>
> *Product Filters:* `ExampleProduct1,ExampleProduct5-ExampleProduct9,"Example-Product1","Example-Product5"-"Example-Product9"`
>
> Equivalent OData: Product eq 'ExampleProduct1' or Product ge 'ExampleProduct5' and Product le 'ExampleProduct9' or Product eq 'Example- Product1' or Product ge 'Example-Product5' and Product le 'Example-Product9'

> ### Note
>
> The comparison operator eq/ge/le in each clause of the resulting OData filter is a lexicographical string comparison. That means that it evaluates both sides from left to right, character by character, treating numbers as ordinary characters in the process. It treats character '2' greater than character '1' even if that character '1' is followed by a character '0'. As a result, the string '2' is greater than the string '10' and so the following filter:
>
> `ExampleEntry2-ExampleEntry10`
>
> doesn't return any data. To receive the desired data in this particular example, you could split the filter into two clauses as follows:
>
> `ExampleEntry2-ExampleEntry9,ExampleEntry10`

3.5 Working with Extensions

Parameters for field extensibility allow you to specify additional attribute mappings 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.

3.5.1 Working with Field Extensions

With field extensions, you can specify the field mapping for the additional attributes that you integrate, or you can change existing data mapping.

In general, the required syntax of the value of the `Field Extensions` parameter is XSLT as follows:

Sample Code

```
<FIELDNAME value = "..." skip= "CONDITION WHEN TO SET ATTRIBUTE VALUE TO NULL
IN SAP IBP" />
```

Note

The `FIELDNAME` must be a field that is listed in the `Attributes in SAP IBP` parameter.

The `"..."` can be defined as a constant value, such as `"0"` or `"TEXT"`. If you use a constant value, all the rows are filled with this value for the given field. You can also define the `"..."` as a function mixed with an XPath expression. This way, you can select specific values from the data set or define a logic using exact values.

The entered value of `skip` is evaluated as either true or false. If it's evaluated `true`, the attribute tag is not added to the output. The result of that is that the attribute value in SAP IBP will be NULL for this row. Alternatively, you can use the `nil` keyword with the same syntax instead of `skip` to set the attribute value to NULL.

Note

If you read a new source field and don't provide a mapping in the field extension parameter, the field value will not be integrated into SAP IBP. Also, if you add a new attribute in the `Attributes in SAP IBP` parameter and don't provide the mapping here, the value for this field will be integrated as NULL.

Example

Using the following code, you can define `CUSTOMFIELD` to be `CustomfieldXXX` where `XXX` is the ID of the corresponding row in the data set:

```
<CUSTOMFIELD value = "concat('Customfield',./ID)">
```

Example

Using the following code, you can skip the field for a certain ID value:

```
<CUSTOMFIELD value = "./DESIREDS4FIELD" skip = "ID='ID value'">
```

You can define an evaluation like the above for any of the fields and with different logical functions. Operations such as `FIELD != ''` also work.

Note

The value of the DESIREDS4FIELD can be any of the fields that are requested from SAP S/4HANA Cloud Public Edition. In the CDS view, you can check which fields are included in the request. You cannot extend the list of the fields in the request, however, you can cycle through the values of the data set using an XPath expression.

In general, the data structure of an XPath expression looks as follows:

Sample Code

```
<item>
<field1>value1a</field1>
<field2>value1b</field2>
</item>
<item>
<field1>value2a</field1>
<field2>value2b</field2>
</item>
…
```

Based on the above sample, to select `value1a` and `value2a`, use `./field1`, and to select `value1b` and `value2b`, use `./field2`.

Note that the structure of the data can be different at this stage, therefore, it is recommended to always check the structure of the data set before executing the XPath selection.

Note

Although the integration flow validates the syntax of the field extension XML, you need to make sure that its content is defined according to your business needs.

3.5.1.1 Extending Plant Data

Sample Code

```
<LOCIDDISPLAY

                        value="concat('Plant: ',Plant)"
skip="not(Plant=('1020','1000','1010'))" nil="Plant='1010'"/>

                        <LOCBUPAID

                        value="concat('BuPa of plant ',./Plant)"/>

                        <LOCREGION value="if(./Plant='1010')then
concat('region of ',./Plant)

                        else 'other region'"/>
```

The above code makes the following modifications:

- Excludes plants with the IDs `1020`, `1000`, and `1010` from the integration.
- If the plant ID is `1010`, it is replaced with null.
- If the location business partner ID (`LOCBUPAID`) becomes `BuPa of plant <plant ID>`.
- The region of the plant with the ID `1010` becomes `region of 1010`, while the region values of all other plants become `other region`.

In this case, the following output is generated based on the parameter value:

```
<LOCID>1000</LOCID>
                    <LOCIDDISPLAY>Plant: 1000</LOCIDDISPLAY>
                    <LOCBUPAID>BuPa of Plant 1000</LOCBUDPAID>
                    <LOCREGION>other region<LOCREGION>
                    <LOCID>1010</LOCID>
                    <LOCIDDISPLAY></LOCIDDISPLAY>
                    <LOCBUPAID>BuPa of Plant 1010</LOCBUDPAID>
                    <LOCREGION>region of 1010<LOCREGION>
                    <LOCID>1020</LOCID>
                    <LOCIDDISPLAY>Plant: 1020</LOCIDDISPLAY>
                    <LOCBUPAID>BuPa of Plant 1020</LOCBUDPAID>
                    <LOCREGION>other region<LOCREGION>
                    <LOCID>1030</LOCID>
                    <LOCBUPAID>BuPa of Plant 1030</LOCBUDPAID>
                    <LOCREGION>other region<LOCREGION>
```

3.6 Scheduling the Integration Flow

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

By default, the start of the integration is scheduled for 2100-01-01 to prevent unnecessary integration jobs during the initial deployment. After you've finalized the configuration of the integration flow, you can manually set the timer according to your needs.

You can select *Run Once* to start integration directly. You can also schedule the job for a future date or make it recurring. For more information about scheduling, see Define a Timer Start Event.

4 Troubleshooting

The following examples of errors and their possible solutions might help you troubleshoot your integration process:

- **Integration fails after successful batch creation**

  Output Code

  ```
  org.apache.camel.RuntimeCamelException:
  com.sap.gateway.core.ip.component.exceptions.OData4Exception: <base
  URL>/sap/opu/odata4/sap/cdi_cds/cdi_cds/sap/i_plant/0001/, cause:
  java.net.SocketTimeoutException: Read timed out (local port <port number>
  to address <IP address> (<GUID>), remote port <port number> to address <IP
  address>)
  ```

  To resolve this issue, try again later or check your SAP S/4HANA system for issues.

- **The certificate in use has expired**

  Output Code

  ```
  org.apache.camel.RuntimeCamelException:
  com.sap.gateway.core.ip.component.exceptions.OData4Exception: Received
  fatal alert: certificate_expired, cause:
  javax.net.ssl.SSLHandshakeException: Received fatal alert:
  certificate_expired
  ```

  To resolve this issue, renew the certificate.

- **Fields are missing from the `Attributes in SAP IBP` parameter, which are included in the `Field Extensions` parameter**

  Output Code

  ```
  javax.script.ScriptException: java.lang.Exception: java.lang.Exception:
  Fields are missing from the Attributes in SAP IBP parameter which
  are included in the Field Extensions parameter. Missing fields:
  [GEOLONGITUDE]@ line 56 in Validations.groovy, cause: java.lang.Exception:
  Fields are missing from the Attributes in SAP IBP parameter which are
  included in the Field Extensions parameter. Missing fields: [GEOLONGITUDE]
  ```

  To resolve the issue, align the value of the `Attributes in SAP IBP` parameter with the mapping defined in the `Field Extensions` parameter.

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

THE BEST RUN SAP