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

Integrating Sales Order History 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

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 14] 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. Working with Source Fields from SAP S/4HANA Cloud Public Edition Parameter [page 18] chapter has been added. Restructured the Working with Extensions [page 16] chapter.
1.1	May 15, 2024	Parameters in Defining Additional Parameters [page 8] have been updated. Time Aggregation [page 13] chapter has been added.
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.

The *Integrate Sales Order History Data from SAP S/4HANA Cloud Public Edition to SAP IBP* integration flow collects sales order data from SAP S/4HANA Cloud Public Edition, and transfers it into an existing location product 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 SAP S/4HANA Cloud Public Edition to SAP IBP* integration flow with the Process Direct connection type, use the **Integrate_Sales_Order_History_Data_from_SAP_S4HANA_Cloud_to_SAP_IBP** address.

2 Prerequisites

- You have run the *Integrate Products from SAP S/4HANA Cloud Public Edition to SAP IBP* integration flow. The *Integrate Sales Order History Data from SAP S/4HANA Cloud Public Edition to SAP IBP* integration flow uses a filter that is saved in a datastore created by the *Integrate Products from SAP S/4HANA Cloud Public Edition to SAP IBP* integration flow, and if the datastore is not available, then all sales order history data is integrated from SAP S/4HANA Cloud Public Edition.
- 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

You can map the fields of the OData API to attributes in SAP Integrated Business Planning (SAP IBP) for data integration.

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

Field in OData API	Field in SAP IBP	Further Hints
`Product`	`PRDID`	
`Plant`	`LOCID`	
`SoldToParty`, `ShipToParty`, or a dummy customer	`CUSTID`	
Period type	`KEYFIGUREDATE`	The field in the OData API depends on the value of the `Time Period Type in SAP S/4HANA Cloud Public Edition` parameter as follows: Week or calendar week: `FirstDayOfWeekDate` Month: `FirstDayOfMonthDate` Date: `DeliveryDate` Technical week: `FirstDayOfWeekDate` or `FirstDayOfMonthDate`, whichever is later

<table>
<tr><th>Field in OData API</th><th>Field in SAP IBP</th><th>Further Hints</th></tr>
<tr><td>Quantity type</td><td>ACTUALSQTY</td><td>The field in SAP IBP is the attribute defined in the <code>Key Figure Name</code> parameter. The value is considered in the base unit of measure.

The field in the API depends on the value of the <code>Quantity Type</code> parameter as follows:

Requested: The sum of <code>ScheduleLineOrderQuantity</code>, as <code>sumOfQuantity</code>

Confirmed: The sum of <code>ConfdOrderQtyByMatlAvailCheck</code>, as <code>sumOfQuantity</code>

Delivered: The sum of <code>DeliveredQuantityInBaseUnit</code>, as <code>sumOfQuantity</code></td></tr>
</table>

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`	`PRDID,LOCID,CUSTID`	Define the attributes of the sales order history master data type to which you want to upload data. **Note** If an attribute is removed from the parameter's default vales, it's also removed from the integrated attributes. KEYFIGUREDATE attribute is always passed to SAP IBP in the attribute list, even if it's not included here.
`Batch Name`	`BatchName`	`Sales Order History Run ID: ${header.SAP_MplCorrelationId}`	Define the name of the data batch. This name also identifies the corresponding job in the *Data Integration Jobs* app.
`Customer Filter`	`CustomerFilter`	`-keep default-`	Optionally, select customers by defining specific customer IDs or intervals. Separate data by commas. **Example** `10100001-10100003,99999999` If you are using a dummy customer, this filter is not used.

Parameter Name	Parameter ID	Default Value	How to Configure the Parameter?
`Customer Source for SAP IBP`	`CustomerSourceforSAPIBP`	`DummyCustomerID`	Define the attribute in SAP S/4HANA Cloud Public Edition that you want to map to the `CUSTID` field in SAP IBP. If you don't want to aggregate data based on customers, you can define a dummy customer, and data is then taken from the `Dummy Customer ID` attribute in the *Define Default Values for Data Integration Between SAP IBP and SAP S/4HANA Cloud Public Edition* integration flow. You can enter one of the following values: • `SoldToParty` • `ShipToParty` • `DummyCustomerID`
`Datastore ID for Product Plant Filter`	`DatastoreIDforProductPlantFilter`	`-keep default-`	Optionally, define the datastore to be used for further filtering data. This filtering ensures that only existing product plant combinations are integrated. If you leave it empty, nothing is filtered out.
`Date From`	`DateFrom`	`xsd:yearMonthDuration('-P2Y') + xsd:date(substring($IFlowStartTimestamp,1,10))`	Define the starting date of the data to be integrated from SAP S/4HANA Cloud Public Edition. The default starting date is two years before the current date. Use the date format **`"yyyy-mm-dd"`**. You can also use XPath expressions when configuring the integration flow.

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 SAP S/4HANA Cloud Public Edition. The default last date is the current date. Use the date format **"yyyy-mm-dd"**. You can also use XPath expressions when configuring the integration flow.
Destination for SAP IBP	DestinationforSAPIBP	-keep default-	Enter the name of the SAP IBP system to which data is transferred.
Field Extensions	FieldExtensions		Optionally, overwrite the default mapping for the predefined fields or specify field mapping for newly added fields. Mapping should be expressed using XSLT.
Further Filters	FurtherFilters		Optionally, define additional filters for planned independent requirements.
Host for SAP S/4HANA Cloud Public Edition	HostforSAPS4HANACloud	-keep default-	Define the base URL of the SAP S/4HANA Cloud API
Key Figure Name	KeyfigureName	ACTUALSQTY	Enter the name of the key figure in SAP IBP into which you want to integrate sales order history data from SAP S/4HANA Cloud Public Edition.
OData Package Size	ODataPackageSize	50000	Define the number of values to be integrated in one data package. The entered value must be 5000 or higher.
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.

Parameter Name	Parameter ID	Default Value	How to Configure the Parameter?
`Plant Filter`	`PlantFilter`	`-keep default-`	Optionally, select plants by defining specific plant IDs or intervals. Separate data by commas. **Example** `0001,0002-9000,9999`
`Product Filter`	`ProductFilter`	`-keep default-`	Optionally, select products by defining specific product IDs or intervals. Separate data by commas. **Example** `FG126,TG11_HD_001,TG10_IBP` Note that only closed intervals are supported.
`Source Fields from SAP S/4HANA Cloud Public Edition`	`SourceFieldsfromSAPS4HANACloud`	`Product,Plant,CustomerSourceforSAPIBP`	Define the sales order attributes that you want to retrieve from the OData service. For further information, see Working with Source Fields from SAP S/4HANA Cloud Public Edition Parameter [page 18].
`Quantity Type`	`QuantityType`	`Requested`	Define the type of sales order history data. You can enter the following values: • `Requested` • `Confirmed` • `Delivered`

Parameter Name	Parameter ID	Default Value	How to Configure the Parameter?
`Time Period Type in SAP S/4HANA Cloud Public Edition`	`TimePeriodTypeinSAPS4HANACloud`	`Technical week`	Define the time aggregation level for the data integrated from SAP S/4HANA Cloud Public Edition. You can use one of the following values: • `Date` • `Week` • `Technical Week` • `Month`
`Time Profile Level in SAP IBP`	`TimeProfileLevelinSAPIBP`	`2`	Define the time disaggregation level to be used in SAP IBP. You can check your time profile settings in SAP IBP using the *Planning Areas* or the *Time Profiles* app.

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 Time Aggregation

You can configure `Time Profile Level in SAP IBP` and `Time Period Type in SAP S/4HANA Cloud Public Edition` parameters. By matching the time levels, you can process data without aggregation attempts.

To load values into SAP IBP, you have the option to configure parameters such as the `Time Profile Level in SAP IBP` and the `Time Period Type in SAP S/4HANA Cloud Public Edition`. 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. If you upload data on a time profile level that's lower than the base time profile level of the key figure, this 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. Assuming that time profile level **4** means month in SAP IBP, time profile level **2** means calendar week, and the base time profile level of the loaded key figure is calendar week as well, you can do the following:

> ### Example
>
> **Time Period Type: `Week`**

Time Profile Level: `2`

Alternatively, you can enable disaggregation as follows:

Example

Time Period Type: `Month`

Time Profile Level: `4`

Note

Disaggregation happens in SAP IBP rather than in SAP Cloud Integration. Therefore, ensure the time period time and the time profile level values match in SAP Cloud Integration even during disaggregation.

3.5 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`

3.6 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.

3.7 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.

3.7.1 Working with Field Extension

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.

3.7.2 Working with Source Fields from SAP S/4HANA Cloud Public Edition Parameter

Source Fields from SAP S/4HANA Cloud Public Edition parameter defines the fields that are requested from the OData service, and determines the `orderby` and `groupby` OData query options. The values are grouped based on the order in which they are entered into the parameter.

You can request or refer to attributes that are available in the Sales Order OData service. Along with the attributes that are provided by the OData service, you can also use `CustomerSourceforSAPIBP` field as a reference to the customer source of your choice according to that parameter.

> **Example**
>
> If the *Customer Source for SAP IBP* parameter is set to `ShipToParty`, the `CustomerSourceforSAPIBP` value in the *Source Fields from SAP S/4HANA Cloud Public Edition* is replaced with `ShipToParty`.
>
> If the *Customer Source for SAP IBP* parameter is set to `DummyCustomerID`, no customer field is included in the *Source Fields from SAP S/4HANA Cloud Public Edition* parameter.
>
> Clearing the field disables the corresponding filtering. If you clear the *Customer Source for SAP IBP* field, the `Customer` value isn't included in the `CustomerSourceforSAPIBP` parameter.

> **Note**
>
> The *Date* and *Quantity* fields are always included in the `orderby` and `groupby` OData query options, even if they are not included in this parameter.
>
> The parameter is case-sensitive. If it's set to empty, it uses the default value (`Product,Plant,CustomerSourceforSAPIBP`) for further information, see Defining Additional Parameters [page 8].
>
> If there are any duplicate values, only the first value is retained, for example:
>
> - Parameter value: `Product,Plant,CustomerSourceforSAPIBP,Product`
> - `orderby/groupby` value: `Product,Plant,CustomerSourceforSAPIBP`

3.7.2.1 Extending Sales Order History Data

Use the following parameters to extend data:

- *Attributes in SAP IBP*
- *Field Extensions*
- *Source Fields from SAP S/4HANA Cloud Public Edition*

The *Source Fields from SAP S/4HANA Cloud Public Edition* parameter is used for data extension. This parameter determines the fields within the Sales Order OData service to be selected for further use.

The *Attributes in SAP IBP* parameter defines the target attributes that receive data in SAP IBP. This also limits the list of fields available for mapping via the *Field Extensions* parameter. Only fields listed in the *Attributes in*

SAP IBP field can be mapped correctly. Fields that are not listed can't be mapped correctly, regardless of the syntactic accuracy of the mapping string.

The *Field Extensions* parameter defines the mapping between the source and the target values. For further information, see Working with Field Extension [page 16].

Example

In the following example, an extra `SALESORG` attribute is integrated in each parameter:

> ### Example
>
> *Attributes in SAP IBP:* `PRDID,LOCID,CUSTID,SALESORG`
>
> *Field Extensions:* `<SALESORG value= "./SalesOrganization/text()" />`
>
> *Key Figure Name:* `EXTENSIONEXAMPLE`
>
> *Source Fields from SAP S/4HANA Cloud Public Edition:*
> `Product,Plant,CustomerSourceforSAPIBP,SalesOrganization`

This example shows that the integration includes the `EXTENSIONEXAMPLE` key figure. Integration to a certain planning level requires all the root attributes of that planning level and at least on key figure value.

In addition to the default values, the integration also requests *SalesOrganization* from the OData service. Using the mapping defined in the *Field Extensions*, integrates data into the `SALESORG` attribute.

The *Field Extensions* parameter defines that attribute's content using the value property. In the example, the *SalesOrganization* field provides the value for the attribute.

Example

The following example sets the `ACTUALSQTY` key figure to NULL in SAP IBP and adds the `DELIVQTY` field that uses the value of `sumOfQuantity`, which is an inner variable for aggregating quantities.

> ### Example
>
> *Attributes in SAP IBP:* `PRDID,LOCID,CUSTID,ACTUALSQTY,DELIVQTY`
>
> *Field Extensions:* `<ACTUALSQTY skip="true"/><DELIVQTY value="sumOfQuantity/text()"/>`
>
> *Key Figure Name:* `DELIVQTY`

> ### Note
>
> You can filter for the extra properties in the *Further Filters* parameter.
>
> On the IBP side, you need additional configurations to the planning area. The key figure needs to have a planning level that includes the attributes set in the *Attributes in SAP IBP* parameter to receive appropriate data.

3.7.2.2 Reducing Sales Order History Data

In the following example, the attribute `CUSTID` is removed:

> **Example**
>
> *Attributes in SAP IBP:* `PRDID,LOCID`
>
> *Key Figure Name:* `REDUCTIONEXAMPLE`
>
> *Source Fields from SAP S/4HANA Cloud Public Edition:* `Product,Plant`

This example shows that the integration includes the `REDUCTIONEXAMPLE` key figure. Integration to a certain planning level requires all the root attributes of that planning level and at least on key figure value.

The integration doesn't request the *Customer* field from SAP S/4HANA. We integrate the remaining data into a key figure with a planning level that doesn't include a `CUSTID` attribute. Data will be aggregated to `Product`, `Plant`, and `Date`.

> **Note**
>
> When you integrate to a planning level where *Product* or *Plant* is not used, you shouldn't include them in the *Source Fields from SAP S/4HANA Cloud Public Edition* parameter. In this case, the *Datastore ID for Product Plant Filter* parameter needs to be blank, otherwise the filter will remove every entry from the response.

4 Troubleshooting

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

- No data is found in the given source. In this case, the integration flow escalates.
- The defined filters are invalid.

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

- The value defined in the `OData Package Size` is smaller than the number of entries of a single product.
- A product plant combination is already deleted in SAP IBP, but the datastore can still contain this combination so data integration is attempted.
- You are trying to integrate data to different planning levels using the `Field Extensions` parameter, however, you can only integrate to one planning level.
- The format of the OData request is invalid. In this case, the error message looks like the following:

  ```
  org.apache.camel.RuntimeCamelException:
  com.sap.gateway.core.ip.component.exceptions.OData4Exception: (/IWCOR/
  CX_OD_EXPR_SYNTAX_ERROR/<GUID>) Invalid token at position
  13null [HTTP/1.1 400 Bad Request], cause:
  org.apache.olingo.client.api.communication.ODataClientErrorException: (/IWCOR/
  CX_OD_EXPR_SYNTAX_ERROR/<GUID>) Invalid token at position 13null [HTTP/1.1 400
  Bad Request]
  ```

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