



Integration Guide | PUBLIC

Document Version: 1.0.0 – 2026-08-01

SAP IBP Read - Read Post Processing Results

Reusable Integration Flow to Retrieve Data Integration Post-Processing Results from SAP Integrated Business Planning in SAP Cloud Integration

Content

1	Document History.	3
2	Introduction.	4
3	Prerequisites.	5
4	How to Call the Integration Flow.	6
5	Configuring the Integration Flow.	8
5.1	Defining Additional Parameters.	11
6	Using the Custom Extension Integration Flow.	13
7	Reading Various Data Types.	15
8	Error Handling.	17
9	Attachments and Monitoring.	18

1 Document History

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

Version	Date	Description
1.0.0	August 1, 2026	Initial version

2 Introduction

You can read post-processing results from SAP Integrated Business Planning (SAP IBP) using the [SAP IBP Read - Read Post Processing Results](#) integration flow (iFlow). You can find this iFlow in the [SAP IBP - Reusable Integration Flows](#) package from SAP IBP 2608 or higher.

If you provide one or more batch IDs, the integration flow retrieves information about what was loaded in those batches. For each file in each batch, it reads the corresponding post-processing results from the SAP IBP reporting table and returns the data as structured XML.

3 Prerequisites

Before you use this integration flow, make sure you have the following:

- You have configured the connection between SAP IBP and SAP Cloud Integration using the SAP_COM_0931 communication scenario (Planning - Integration Suite - Cloud Integration).
- You have a valid RFC destination for your SAP IBP system configured in your SAP BTP subaccount.
- You have *SAP IBP Read - Initialize* integration flow version 1.1.3 or higher working in your system.
- You have existing data integration batches in SAP IBP.

4 How to Call the Integration Flow

The integration flow supports two calling scenarios:

- Scenario A - The integration flow is called directly after a write process. The body of the calling integration flow already contains batch information. You don't need any additional configuration to identify the batch. Accepted write processes:
 - [SAP IBP Write - Inbound Integration Generic Process](#)
 - [SAP IBP Add-On Read - Generic Process](#)
 - [SAP IBP Write - Process Posted Data](#) as part of the [SAP IBP Write Reusable Integration Flows](#). For more information, see [Integration Using SAP IBP - Reusable Integration Flows](#).
- Scenario B - The integration flow is called standalone. The caller provides one or more batch IDs and optional filter parameters through externalized parameters.

To configure the [SAP IBP Read - Read Post Processing Results](#) integration flow (iFlow) from another integration flow, use a [Process Direct](#) call with the process ID `SAP_IBP_Read_-_Read_Post_Processing_Results`.

Scenario A — Called After a Write Process

When you call this integration flow after a write process, the body already contains batch information. The integration flow automatically extracts the batch ID and destination from the body. Three body formats are supported:

❖ Example

Output of [SAP IBP Write - Inbound Integration Generic Process](#):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<IBPInboundIntegrationGenericProcess
AfterDynamicDataLoadExtensionPoint=""
BatchCommand="INSERT_UPDATE"
BatchID="12345"
DestinationforSAPIBP="IBP_DESTINATION"
PlanningArea="SAPIBP1"
PlanningAreaVersion="__BASELINE">
<Messages/>
<ProcessPostedDataResponse
...
</ProcessPostedDataResponse>
</IBPInboundIntegrationGenericProcess>
```

❖ Example

Output of [SAP IBP Add-On Read - Generic Process](#):

```
<?xml version="1.0" encoding="utf-8"?>
<IBPRead Status="Closed"
```

```
BatchID="12345"
DestinationforSAPIBP="IBP_DESTINATION"
PlanningArea="SAPIBP1"
PlanningAreaVersion="__BASELINE">
<Messages/>
</IBPRead>
```

❁ Example

Output of *SAP IBP Write - Process Posted Data*:

```
<?xml version="1.0" encoding="utf-8"?>
<multimap:Messages xmlns:multimap="http://sap.com/xi/XI/SplitAndMerge">
  <multimap:Message1>
    <IBPWriteMasterData Status="OK">
      <Request/>
      <Messages/>
    </IBPWriteMasterData>
  </multimap:Message1>
  <multimap:Message2>
    <IBPWriteBatch Id="12345" Destination="IBP_DESTINATION">
      <Files/>
    </IBPWriteBatch>
  </multimap:Message2>
</multimap:Messages>
```

📌 Note

In Scenario A, only the batch ID and destination are read from the body. Additional filters (`FileName`, `MasterDataType`, `DataType`, and others) can still be applied via the header parameters described in [Defining Additional Parameters \[page 11\]](#).

Scenario B — Called Standalone

When you call the integration flow standalone, using a wrapper iFlow, leave the body empty and provide the batch ID and other parameters through message headers.

5 Configuring the Integration Flow

Procedure

1. Before you call the integration flow, set the required parameters either in the message body (Scenario A) or as message headers (Scenario B). See [Defining Additional Parameters \[page 11\]](#) for the full list.
2. Configure the connection to the integration flow using a *Request Reply* step with a *ProcessDirect* receiver adapter.
3. Set the *Address* on the *Connection* tab to:

SAP_IBP_Read_-_Read_Post_Processing_Results

Results

Once the integration flow completes, the result is available in the message body as XML in the following structure:

❖ Example

```
<Batches>
  <Batch ID="12345">
    <Files>
      <File name="CUSTOMER.csv">
        <IBPReadMasterData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
          <item>
            <BATCH>12345</BATCH>
            <CUSTID>C001</CUSTID>
            <CUSTDESCR>Customer 1</CUSTDESCR>
            <MSGID>0</MSGID>
            <S_MSGCLASS xsi:nil="true"/>
            <MSGV1 xsi:nil="true"/>
            <MSGV2 xsi:nil="true"/>
            <S_MSGTYPE xsi:nil="true"/>
          </item>
        </IBPReadMasterData>
      </File>
      <File name="ACTUALSQTY.csv">
        <IBPReadKeyFigures xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
          <NoData>No data returned</NoData>
        </IBPReadKeyFigures>
      </File>
    </Files>
  </Batch>
</Batches>
```

If the integration flow finds no data for any batches, or all batches fail, the output is: <Batches/>

The integration flow returns validation error codes, not text.

❖ Example

XSD schema of the output structure:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="unqualified"
  attributeFormDefault="unqualified">

  <!-- ===== -->
  <!-- Root element -->
  <!-- ===== -->
  <xs:element name="Batches">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Batch" type="BatchType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <!-- ===== -->
  <!-- Batch – one per IBP batch ID processed -->
  <!-- ===== -->
  <xs:complexType name="BatchType">
    <xs:sequence>
      <xs:element name="Files" type="FileType"/>
    </xs:sequence>
    <xs:attribute name="ID" type="xs:string" use="required"/>
  </xs:complexType>

  <!-- ===== -->
  <!-- Files – container for all files belonging to the batch -->
  <!-- ===== -->
  <xs:complexType name="FileType">
    <xs:sequence>
      <xs:element name="File" type="FileType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!-- ===== -->
  <!-- File – one per (batch, file) tuple that produced a result -->
  <!-- Contains exactly one of: -->
  <!-- - IBPReadMasterData (when type="MASTERDATA") -->
  <!-- - IBPReadKeyFigures (when type="KEYFIGURE") -->
  <!-- ===== -->
  <xs:complexType name="FileType">
    <xs:choice>
      <xs:element name="IBPReadMasterData" type="ReadResultType"/>
      <xs:element name="IBPReadKeyFigures" type="ReadResultType"/>
    </xs:choice>
  </xs:complexType>
</xs:schema>
```

```

</xs:choice>
<xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>

<!-- ===== -->
<!-- ReadResultType -->
<!-- Either: -->
<!-- - one or more <item> elements with open content, OR -->
<!-- - a single <NoData> element -->
<!-- ===== -->
<xs:complexType name="ReadResultType">
<xs:choice>
<xs:element name="item" type="ItemType" minOccurs="1" maxOccurs="unbounded"/>
<xs:element name="NoData" type="xs:string"/>
</xs:choice>
</xs:complexType>

<!-- ===== -->
<!-- ItemType – open content -->
<!-- The actual fields are driven by the runtime Select parameter -->
<!-- (BATCH, the requested key fields, MSGID, S_MSGCLASS, MSGV1, -->
<!-- MSGV2, S_MSGTYPE). Each child element holds either a string -->
<!-- value or a self-closing tag with xsi:nil="true" for nulls. -->
<!-- ===== -->
<xs:complexType name="ItemType">
<xs:sequence>
<xs:any namespace="##any" processContents="skip"
minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

</xs:schema>

```

The Message Processing Log (MPL) status is one of the following:

Status	Meaning
COMPLETED	All batches were read successfully.
ESCALATED	One or more batches could not be read. Results from successful batches are still included in the output.
ERROR	A technical error occurred (for example, invalid parameters or a connection failure). No results are returned.

5.1 Defining Additional Parameters

You can configure the integration flow (iFlow) by setting parameters. When you call it from a wrapper integration flow using [Process Direct](#), use the parameter ID.

Parameters

Parameter Name	Parameter ID	Default Value	How to Configure the Parameter
Batch IDs	BatchIDs		Enter the ID of the batch to read. For standalone use, you can provide multiple batch IDs as a comma-separated list (for example, 12345, 67890). In Scenario A, the iFlow takes the data integration batch ID from the message body. This parameter is mandatory.
Custom Extension Integration Flow Address	CustomExtensionIntegrationFlowAddress		Optionally, enter the ProcessDirect address of a custom extension integration flow. When set, enables packaging mode for large datasets - each package is forwarded to the specified integration flow.
Data Type	DataType		Optionally, filter results by data type. Accepted values: MASTERDATA, KEYFIGURE. If you leave it empty, both types are returned.  Restriction CALENDAR data is not supported and is excluded from results.
Destination for SAP IBP	DestinationforSAPIBP		Enter the RFC destination name for your SAP IBP system, as you configured in your SAP BTP subaccount. In Scenario A, the iFlow takes the destination from the message body. This parameter is mandatory.
File Name	FileName		Optionally, filter results by specific file names. You can filter multiple file names by separating them with commas, for example, CUSTOMER.csv, PRODUCT.csv . If you leave it empty, the iFlow returns all files in the batch.
Log Into Attachment	LogIntoAttachment	true	When set to true , the full output is saved as an MPL attachment for monitoring purposes.
Master Data Type	MasterDataType		Optionally, filter results by a specific master data type, for example, IBPCUSTOMER . You can enter multiple values as a comma-separated list, for example, IBPCUSTOMER, IBPLOCATION .

Parameter Name	Parameter ID	Default Value	How to Configure the Parameter
Package Size in Rows	PackageSizeInRows	50000	Define the number of rows per fetch package. Must be a number greater than 0. This parameter must have a value.
Planning Area	PlanningArea		Optionally, filter results by a specific planning area. Enter a single value only.
Planning Area Version	PlanningAreaVersion		Optionally, filter results by a specific planning area version. Enter a single value only.
Read Limit	ReadLimit		Set the maximum number of rows to read per file. It must be a number greater than 0. If empty, all rows are returned.
Return Only Errors	ReturnOnlyErrors	false	When set to true , only rows where the message ID (MSGID) is not 0 are included in the output. Filtering is applied at the SAP IBP query level.

6 Using the Custom Extension Integration Flow

By default, the integration flow accumulates all fetched data in memory and returns it as a single XML response in the message body. For large datasets, this can cause memory issues or message size limits to be exceeded.

To handle large datasets, you can provide a custom extension integration flow via the `CustomExtensionIntegrationFlowAddress` parameter. When set, the integration flow switches to packaging mode: instead of accumulating results in memory, each fetched package is forwarded directly to the extension integration flow for processing. The extension integration flow is responsible for handling the data.

How Packaging Works

The integration flow fetches data from SAP IBP in packages. The size of each package is controlled by the `PackageSizeInRows` parameter (default: 50,000 rows). For each file in each batch:

1. The integration flow calls the [SAP IBP Read - Initialize](#).
2. It fetches data in packages of `PackageSizeInRows` rows until all data is retrieved or the `ReadLimit` is reached.
3. After all packages are fetched, the [SAP IBP Read - Close](#) is called.

❖ Example

A sample package body that a customer extension flow might receive (`PackageSizeInRows: 2`):

```
<Batches>
<Batch ID="12345">
<Files>
<File name="CUSTOMER">
<IBPReadMasterData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<item>
<BATCH>12345</BATCH>
<CUSTID>C001</CUSTID>
<CUSTDESCR>Customer 1</CUSTDESCR>
<CUSTCHANNEL>RT</CUSTCHANNEL>
<CUSTCOUNTRY>US</CUSTCOUNTRY>
<CUSTREGION>NA</CUSTREGION>
<MSGID>0</MSGID>
<S_MSGCLASS xsi:nil="true"/>
<MSGV1 xsi:nil="true"/>
<MSGV2 xsi:nil="true"/>
<S_MSGTYPE xsi:nil="true"/>
</item>
<item>
<BATCH>12345</BATCH>
```

```

<CUSTID>C002</CUSTID>
<CUSTDESCR>Customer 2</CUSTDESCR>
<CUSTCHANNEL>WH</CUSTCHANNEL>
<CUSTCOUNTRY>DE</CUSTCOUNTRY>
<CUSTREGION>EU</CUSTREGION>
<MSGID>0</MSGID>
<S_MSGCLASS xsi:nil="true"/>
<MSGV1 xsi:nil="true"/>
<MSGV2 xsi:nil="true"/>
<S_MSGTYPE xsi:nil="true"/>
</item>
</IBPReadMasterData>
</File>
</Files>
</Batch>
</Batches>

```

Configuring the Extension Integration Flow

Set the `CustomExtensionIntegrationFlowAddress` parameter for the [ProcessDirect](#) address of your extension integration flow. The extension integration flow must be reachable via [ProcessDirect](#) from the same SAP Cloud Integration tenant.

Each package is forwarded as a separate [ProcessDirect](#) call. The message body contains the fetched XML for that package.

The **Reporting Results** MPL attachment is not created, even if `LogIntoAttachment` is set to **true**. The extension integration flow receives the data directly and is responsible for any persistence or logging.

If no data is available, the extension integration flow is not called.

7 Reading Various Data Types

Master Data

To read master data for a specific batch, provide the batch ID and optionally filter by `MasterDataType` or `FileName`.

❖ Example

Batch IDs: 12345

Destination for IBP: IBP_DESTINATION

Master Data Type: CIACUSTOMER

Key Figure Data

To read key figure data, provide the batch ID and optionally the `PlanningArea` and `PlanningAreaVersion`.

❖ Example

Batch IDs: 12345

Destination for IBP: IBP_DESTINATION

Data Type: KEYFIGURE

Planning Area: SAPIBP1

Planning Area Version: __BASELINE

Multiple Batches

To read multiple batches in a single call (Scenario B), provide a comma-separated list of batch IDs. Each batch is processed individually with the same filter settings.

❖ Example

Batch IDs: 12345, 67890, 11111

Destination for IBP: IBP_DESTINATION

Filtering Results

You can combine multiple filter parameters to narrow down the results. Only matching records are returned.

If a filter parameter is empty, that filter is not applied.

`FileName` and `MasterDataType` accept comma-separated values for multi-value filtering.

`DataType` accepts a single value: **MASTERDATA** or **KEYFIGURE**.

`PlanningArea` and `PlanningAreaVersion` accept a single value.

8 Error Handling

Situation	Behavior	MPL Status
Empty or invalid Batch IDs (non-numeric)	Integration flow throws a <code>RuntimeException</code> . No output is returned.	ERROR
Invalid Data Type value	Integration flow throws a <code>RuntimeException</code> . No output is returned.	ERROR
<code>ReadLimit</code> or <code>PackageSizeInRows</code> is 0 or non-numeric	Integration flow throws a <code>RuntimeException</code> . No output is returned.	ERROR
<code>PackageSizeInRows</code> is empty	Integration flow throws a <code>RuntimeException</code> . No output is returned.	ERROR
Invalid RFC destination	Connection error. No output is returned.	ERROR
Batch not found in SAP IBP	Batch is skipped. Other batches continue to be processed.	ESCALATED
Batch Overview returns no matching data for applied filters	Batch is skipped without error. Not counted as a failure.	COMPLETED (if all other batches succeed)
Fetch returns no data with <code>ReturnOnlyErrors=false</code>	<code><NoData>No data returned</NoData></code> is written to the output for that file. Not counted as a failure.	COMPLETED
Fetch returns no data with <code>ReturnOnlyErrors=true</code>	File entry is skipped entirely. Not counted as a failure.	COMPLETED
One or more batches fail, others succeed	Partial output is returned for successful batches.	ESCALATED

9 Attachments and Monitoring

After the integration flow (iFlow) completes, you can view results and logs in [Monitor Message Processing](#) in the SAP Integration Suite.

Attachment	Description
Reporting Results	The full XML result (<Batches>...</Batches>). Only present when <code>LogIntoAttachment</code> is set to true and <code>CustomExtensionIntegrationFlowAddress</code> is set to empty.
Batch Overview {status} {batchId}	Added when a batch overview call returns an error. Contains batch ID, status, and error messages from SAP IBP.
ReadError {batchId}_{fileName}	Added when a read operation for a file returns an error status from SAP IBP. Contains batch ID, target object, file name, status, and error messages from SAP IBP.
Exception	When a technical error occurs, contains the exception message and the message body at the time of failure.

Custom header properties are set on the MPL message for monitoring purposes:

Header Property	Description
Batch {id} Overview	The custom header property Batch {id} Overview is set for every batch with its Batch Overview status (<code>FINISHED</code> , <code>NO_DATA</code> , or <code>ERROR</code>). When the status is an error (neither <code>FINISHED</code> nor <code>NO_DATA</code>), an additional attachment Batch Overview {status} {batchId} is also added containing the batch ID, status, and error messages from SAP IBP.
Read {batchId} {fileName}	Set to <code>NO_DATA</code> when a fetch returns no items (only when <code>ReturnOnlyErrors</code> is false). Also set to the error status when a read call fails.
Error Batch Id	The batch ID being processed when the error occurred. Only set if the error occurs during batch processing.
Error Target Object	The target object being processed when the error occurred. Only set if the error occurs during batch processing.

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.

© 2026 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.

