software AG

ARIS PROCESS MINING

DATA INGESTION API

VERSION 10.0 - SERVICE RELEASE 17

JANUARY 2022

Contents

1 Ingest data using a public API

ARIS Process Mining supports a public data ingestion API. You create and send HTTP requests to use the API. The API allows you to transfer data from any data source to ARIS Process Mining. The data transferred to ARIS Process Mining must be in table format and must conform to JSON format.

You can use an appropriate API client to create HTTP requests and you need the appropriate API programming skills.

The Data transfer objects (DTOs) (page 11) chapter lists the DTOs that you can use for transferring data.

The Endpoints (page 17) chapter lists the endpoints that you can use for your HTTP requests.

PROCEDURE IN ARIS PROCESS MINING

To be able to transfer data to ARIS Process Mining using an API, you must perform the following steps:

- Create a system integration for the data ingestion API (page 2).
- Create a data set to store the transferred data. For details, see the chapter Create a data set.
- Create a connection for the data ingestion API. (page 3)

USE THE DATA INGESTION API TO TRANSFER DATA

The following steps are a best practice for transferring data using an API.

- Authenticate your client (page 5)
- Retrieve, create, or replace source tables (page 6)
- Check if the data set is ready for data upload (page 8)
- Create a data ingestion cycle for the data upload (page 8)
- Upload data for each source table (page 9)
- Start data ingestion (page 9) (Processing of uploaded data on server-side)
- Monitor data ingestion (page 9)
- Check if the data set is ready for data load (page 10)
- Create a data ingestion cycle to start the data load (page 10)
- Monitor data ingestion (data load) (page 10)

1.1 Create a system integration for the data ingestion API

To use the data ingestion API (page 1), you must create a corresponding system integration.

ARIS Process Mining provides the **Client credentials** and **Authorization code** grant types as authentication methods. The data is required to authenticate the API client to ARIS Process Mining.

The **Client credentials** as authentication method uses the technical user for authentication. The authentication is performed in the background. You do not need to log in manually.

If you select the **Authorization code** grant type, you must specify an authorization callback URL. Based on your settings, a well-known URL is created that you can use to obtain the corresponding refresh, authorization, token, and user info endpoints. You need these endpoints for authentication.

Endpoint examples

- refresh_endpoint:
  https://ariscloud.com/umc/api/v1/oauth/refreshtoken?tenant=myprojectroom
- authorization_endpoint:
  https://ariscloud.com/umc/oauthLogin?grant_type=authorization_code&tenant=myprojectroom
- token_endpoint:
  https://ariscloud.com/umc/api/v1/oauth/accesstoken?grant_type=authorization_code&tenant=myprojectroom
- userinfo_endpoint:
  https://ariscloud.com/umc/api/v1/oauth/userinfo?tenant=myprojectroom

Prerequisite

You have installed the ARIS Process Mining Enterprise license.

Procedure

1. Click the ⁝⁝⁝ **Navigation menu** icon -> **Administration** in the program header.
2. Click **System integration** in the **Administration** panel.
3. Click **Add system integration** -> **Data ingestion (API)**. The corresponding dialog opens.
4. Enter a name, for example, Data ingestion, and an optional description.
5. Select an authentication method in the **Grant type (OAuth)** drop-down menu.

   If you select the **Authorization code** grant type, specify the **Authorization callback URL** that is used for authentication.

   https://<host name>/umc/rest/oauth/callback?tenant=<project room>&provider=umc

Replace <host name> with the host name of the ARIS Process Mining installation and the <project room> with the ARIS Process Mining project room you want to login to.

Example

https://ariscloud.com/umc/rest/oauth/callback?tenant=myprojectroom&provider=umc

6. Click **Add**. The **Data ingestion access data** dialog opens. The dialog provides the client ID, secret key, project room name, and additionally a well-known URL if you have selected **Authorization code** as grant type.
7. Save all provided authentication data, for example, using a text editor.
   a. Click **Copy to clipboard**.
   b. Save the key, for example, using a text editor.
8. Open the well-known URL in your browser and save the provided endpoints, for example, using a text editor.
9. Click **Done**.

The system integration is created and listed with the name you specified.

Tip

The access data (except for the endpoints) is saved in the system integration you created. You can display the source system access data to access the client credentials key.

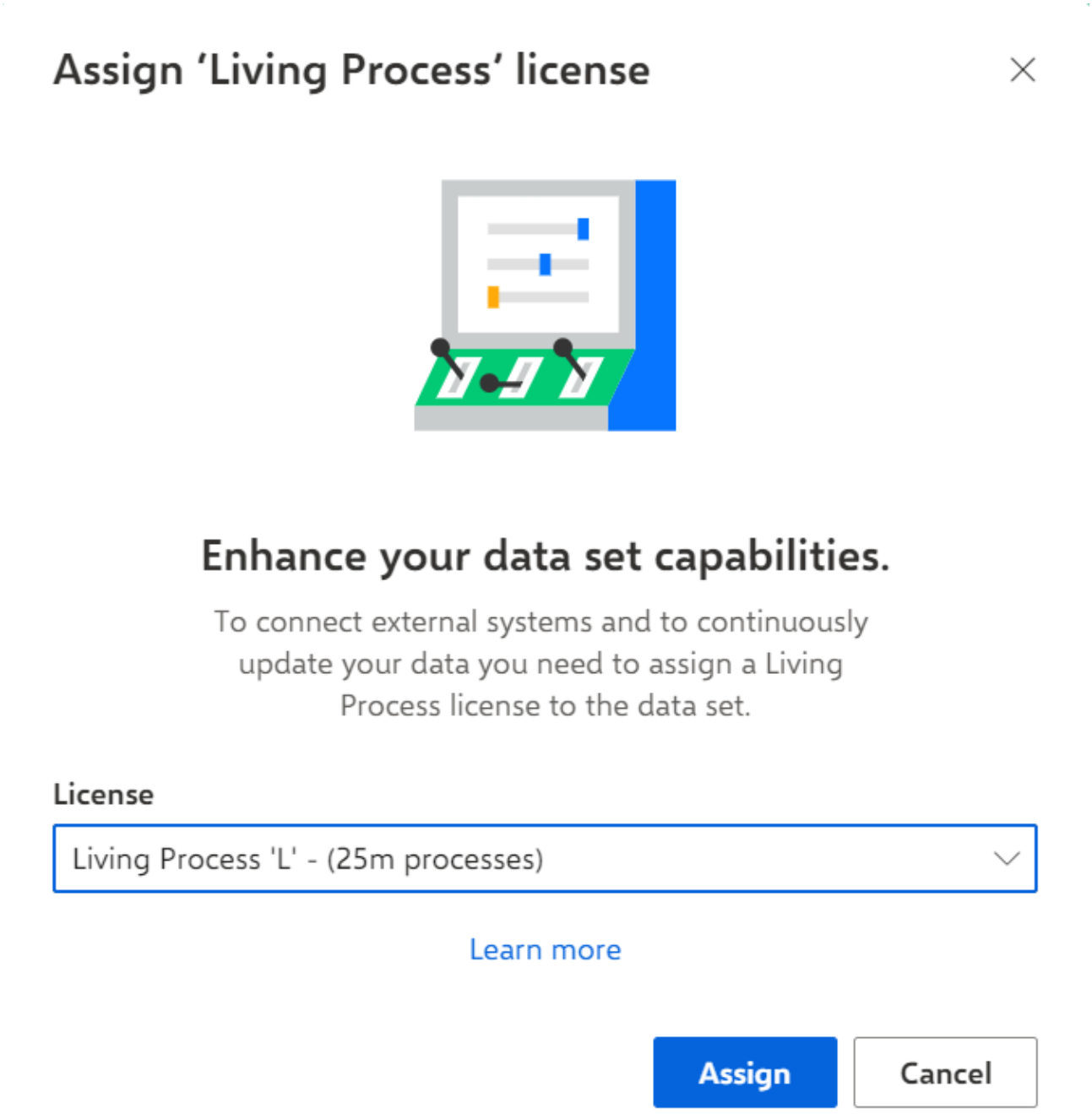
1.2 Create a connection for the data ingestion API

Before you can transfer data to ARIS Process Mining using the data ingestion API, you must create a corresponding connection for the data set where the transferred data is stored. You create a connection to the API client using the created system integration (page 2).

Procedure

1. Open the data set that you want to use for analyzing the source data.
   a. Click the ⁝⁝⁝ **Navigation menu** icon -> ⛁ **Data collection** in the program header.
   b. Click the data set on the **Data sets** page. The selected data set opens.
2. Open the **Connections** component.
3. Click **Add connection**. If you add a connection to a source system for the first time and you have not assigned a 'Living Process' license to the data set yet, the **Assign 'Living Process' license** dialog opens.

4. Select a license in the drop-down menu. You need the ‘Living Process’ license to extract and analyze processes. The number of processes you can extract depends on the selected license.



5. Click **Assign**. The **Add connection** dialog opens.
6. Configure the connection.
   a. Enter a unique name for the connection to the source system, for example, Data ingestion.
   b. Select a system integration created for the data ingestion API.
   c. Click **Add**.

You have created a connection for the API. The created connection is displayed on the **Connections** page with the settings you specified.

1.3 Use the data ingestion API

1.3.1 Authenticate your API client

You must perform an HTTP authentication request to authenticate your client against ARIS Process Mining. Depending on the specified authentication method (page 2), you can use client credentials or an authorization code.

You can find the required data in the system integration created for the data ingestion API (page 2).

AGAINST ARIS CLOUD USING CLIENT CREDENTIALS

Send an HTTP request to the ARIS cloud endpoint and path **/api/applications/login** with the following properties:

- Content type: application/x-www-form-urlencoded
- Query parameters:

  clientId: client ID from the corresponding system integration (page 2)

  clientSecret: client secret from the corresponding system integration (page 2)

  tenant: project room name from the corresponding system integration (page 2)

Ensure that the generated bearer tokens are sent with the appropriate header for every subsequent request.

AGAINST ARIS USER MANAGEMENT USING CLIENT CREDENTIALS

Send an HTTP request to ARIS User Management using the path **/umc/api/oauth/apptoken** with the following properties:

- Content type: application/x-www-form-urlencoded
- Query parameters

  client_id: client ID from the corresponding system integration (page 2)

  client_secret: client secret from the corresponding system integration (page 2)

  tenant: project room name from the corresponding system integration (page 2)

  grant_type: client_credentials

Ensure that generated bearer tokens are sent with the appropriate header for every subsequent request.

USING AUTHORIZATION CODE

Note that your client application must support OAuth 2.0 with **Authorization code** grant type.

Configure the client application to use:

- The callback URL
- Authorization endpoint from the corresponding system integration (page 2)
- Token endpoint from the corresponding system integration (page 2)
- Client ID from the corresponding system integration (page 2)
- Secret key from the corresponding system integration (page 2)
- Client credentials must be sent in the body, not as a Basic Auth header.

Ensure that the generated bearer tokens are sent with the appropriate header for every subsequent request.

1.3.2 Retrieve, create, or replace source tables

To retrieve, create, or replace source tables, perform the following HTTP requests.

RETRIEVE SOURCE TABLES

GET "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/sourceTableDefinitions"

CREATE SOURCE TABLES

POST "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/sourceTables"

Example

```
[
  {
      "name": "table_a",
      "namespace": "default",
      "columns": [
        {
          "dataType": "STRING",
          "name": "column_a1"
        },
        {
          "dataType": "LONG",
          "name": "column_a2"
        },
        {
          "dataType": "DOUBLE",
          "name": "column_a3"
        },
```

```
        {
          "dataType": "FORMATTED_TIMESTAMP",
          "name": "column_a4",
          "format": "yyyy-MM-dd HH:mm:ss.SSS"
        }
      ]
  },
  {
      "name": "table_b",
      "namespace": "default",
      "persistenceMode": "APPEND",
      "columns": [
       ...
      ]
  }
]
```

REPLACE SOURCE TABLES

POST "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/sourceTables?forceReplace=true"

Example

```
[
  {
      "fullyQualifiedName": "default.table_a",
      "columns": [
        {
          "dataType": "STRING",
          "name": "column_a1"
        },
        {
          "dataType": "LONG",
          "name": "column_a2"
        },
        {
          "dataType": "DOUBLE",
          "name": "column_a3"
        },
        {
          "dataType": "FORMATTED_TIMESTAMP",
          "name": "column_a4",
          "format": "yyyy-MM-dd HH:mm:ss.SSS"
        }
      ]
  },
  {
      "fullyQualifiedName": "default.table_b",
      "persistenceMode": "APPEND"
  }
]
```

1.3.3 Check if the data set is ready for data upload

The data set must be ready to upload the data. To check the state of the data set, perform the following HTTP request.

POST "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/readyForIngestion"

Example

```
{
  "dataUploadTargets": [
    {
      "fullyQualifiedName": "default.table_a"
    },
    {
      "fullyQualifiedName": "default.table_b"
    }
  ]
}
```

If the data set is ready, you receive a positive response. Otherwise, the response is negative and contains the corresponding reason.

Example

```
{
  "ready": false,
  "cause": {
    "code": "INR1001",
    "message": "The data set is currently being processed"
  }
}
```

1.3.4 Create a data ingestion cycle for the data upload

If the data set is ready, you can create a data ingestion cycle for the data upload with the following HTTP request.

POST "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/ingestionCycles"

Example

```
{
  "dataUploadTargets": [
    {
      "fullyQualifiedName": "default.table_a"
    },
    {
      "fullyQualifiedName": "default.table_b"
    }
  ]
}
```

The response for the call above returns the fully-formed data ingestion cycle. The initial state is **ACCEPTING_DATA**. All tables referenced by the cycle are locked for everything except the upcoming data upload.

1.3.5 Upload data for each source table

To upload the data for each source table to the data set, perform the following HTTP request.

POST "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/sourceTables/<source table>/data"

Example

```
[
  ["This is a description", 1255, 1385.5, "2021-07-15 18:03:25.889"],
  ["A second example text", 510, -23.58, "2021-07-10 10:59:05.421"],
  ["Example text", 1626347163123, 3.1415, "2021-07-01 08:00:01.002"]
]
```

Larger amounts of data can be uploaded by means of multiple requests. With each request, the data is stored in temporary form on the server.

1.3.6 Start data ingestion

To start the data ingestion, perform the following HTTP request.

PUT "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/ingestionCycles/<ingestion cycle>/dataComplete"

The ingestion cycle state changes to **INGESTING_DATA**. The uploaded temporary data is now persisted in the source database.

1.3.7 Monitor data ingestion (data upload)

To verify that the data upload was successful, perform the following HTTP request.

GET "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/ingestionCycles/<ingestion cycle>/state"

If the data ingestion failed, you receive a response with the corresponding reason:

```
{
  "value": "FAILED",
  "cause": {
    "code": "IER1000",
    "message": "An unexpected error occurred"
  }
```

```
}
```

Otherwise, the ingestion status can be **INGESTING_DATA** if the cycle is still running, **COMPLETED_SUCCESSFULLY** if it went through without any problems, or **CANCELED** if it was aborted (for example, using the API).

1.3.8 Check if the data set is ready for data load

The data set must be ready to load the data into the process storage. To check the state of the data set, perform the following HTTP request.

POST "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/readyForIngestion"

Example

```
{
  "dataLoadTriggered": true
}
```

If the data set is ready, you receive a positive response. Otherwise, the response is negative and contains the corresponding reason

1.3.9 Create a data ingestion cycle to start the data load

If the data set is ready for data load, create a data ingestion cycle to start the data load using the following HTTP request.

POST "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/ingestionCycles"

Example

```
{
  "dataLoadTriggered": true
}
```

The response for the call above returns the fully-formed data ingestion cycle. The initial state is **INGESTING_DATA**. The corresponding data load starts immediately.

1.3.10 Monitor data ingestion (data load)

To verify that the data load was successful, perform the following HTTP request.

GET "https://<host name>/mining/api/pub/dataIngestion/v1/dataSets/<data set>/ingestionCycles/<ingestion cycle>/state"

You receive a response with the current state value (**INGESTING_DATA**, **COMPLETED_SUCCESSFULLY**, or **FAILED**) and optionally a respective reason.

1.4 Data transfer objects (DTOs)

You can use the following data transfer objects (DTOs) for the data ingestion API.

SOURCETABLEDEFINITION

As input

Only in list form, either standalone as presented here or as part of a data ingestion cycle (DataIngestionCycle) (see below). Properties can be mandatory or optional, depending on if a table should be created or replaced.

```
[
    {
        "key": "prq_some_namespace_e",
        "name": "example_table_o",
        "namespace": "some_namespace",
        "fullyQualifiedName":"some_namespace.example_table_o",
        "persistenceMode": "OVERWRITE",
        "columns": [
            {
                "dataType": "DOUBLE",
                "name": "CATEGORY"
            },
            {
                "dataType": "STRING",
                "name": "CATEGORY_NAME"
            },
            {
                "dataType": "FORMATTED_TIMESTAMP",
                "name": "CREATED",
                "format": "yyyy/MM/dd/HH:mm:ss"
            },
            {
                "dataType": "STRING",
                "name": "PROCESSOR"
            },
            {
                "dataType": "STRING",
                "name": "PROCESSOR_GROUP"
            }
        ]
    }
]
```

As output

Only in list form, either standalone as presented here or as part of a data ingestion cycle (DataIngestionCycle) (see below).

```
[
    {
        "key": "prq_some_namespace_e",
        "name": "example_table_o",
        "namespace": "some_namespace",
        "fullyQualifiedName": "some_namespace.example_table_o",
        "persistenceMode": "OVERWRITE",
        "columns": [
            {
                "dataType": "DOUBLE",
                "name": "CATEGORY"
            },
            {
                "dataType": "STRING",
                "name": "CATEGORY_NAME"
            },
            {
                "dataType": "FORMATTED_TIMESTAMP",
                "name": "CREATED",
                "format": "yyyy/MM/dd HH:mm:ss"
            },
            {
                "dataType": "STRING",
                "name": "PROCESSOR"
            },
            {
                "dataType": "STRING",
                "name": "PROCESSOR_GROUP"
            }
        ]
    }
]
```

- Keys will be generated on the server.
- The fully qualified name (fullyQualifiedName) consists of the name and namespace, separated by '.'.
- The persistence mode (persistenceMode) can either be OVERWRITE or APPEND. See chapter Persistence mode (page 21) for more details.
- Columns can be of type DOUBLE, LONG, STRING, and FORMATTED_TIMESTAMP.

DATAINGESTIONREADYSTATE

Used only as output after a readiness check. In case of 'not ready', the ready property is set to 'false' and the object contains a cause with code and message.

```
{
  "ready": false,
```

```
  "cause": {
    "code": "INR1001",
    "message": "The data set is currently being processed"
  }
}
```

Causes consist of a code and a message to indicate why exactly the data set is not ready. The code is four digits long and always prefixed with "INR" for "Ingestion - Not Ready". A list with the concrete codes and the semantics can be found in the table below.

Code	Semantic
INR1000	Undefined. Used for unexpected situations.
INR1001	Data set in process. The referenced data set is locked by another process (data ingestion cycle, manual data load, etc.). The user must wait until the locked is lifted and repeat the request.
INR1002	Living process quota exceeded. Usage of the data ingestion API is limited to data sets with a 'living process' license. The assigned 'living process' license defines an upper bound (process quota) for the number of process instances within the corresponding data set. Code INR1002 indicates that the process quota defined by the assigned 'living process' license is already exceeded and it is not allowed to ingest more data. The user can reduce the number of process instances within the dataset by deleting process instances or alternatively assign a 'living process' license with a higher process quota.
INR1003	Unexpected source table type. Indicates that at least one of the referenced source tables has an unexpected type, for example, was created via the CSV upload. To use a table in the context of the data ingestion API, it either needs to have been created by that API or updated by it via the replace source table functionality.
INR1004	No data to load. Indicates that there is no new pending data for the tables referenced in the data modeling section. Therefore a data load is not necessary.

DATAINGESTIONCYCLE

As input

In case of data upload

```
{
    "dataUploadTargets": [
        {
            "fullyQualifiedName": "some_namespace.example_table_a"
        }
    ]
}
```

In case of data load

```
{
    "dataLoadTriggered": true
}
```

As output

Either in list form

```
[
    {
        "key": "api_2",
        "dataUploadTargets": [
            {
                "key": "prq_some_namespac_38",
                "name": "example_table_a",
                "namespace": "some_namespace",
                "fullyQualifiedName": "some_namespace.example_table_a",
                "persistenceMode": "APPEND",
                "columns": [
                    {
                        "dataType": "DOUBLE",
                        "name": "CATEGORY"
                    },
                    {
                        "dataType": "STRING",
                        "name": "CATEGORY_NAME"
                    },
                    {
                        "dataType": "FORMATTED_TIMESTAMP",
                        "name": "CREATED",
```

```
                    "format": "yyyy/MM/dd HH:mm:ss"
                },
                {
                    "dataType": "STRING",
                    "name": "PROCESSOR"
                },
                {
                    "dataType": "STRING",
                    "name": "PROCESSOR_GROUP"
                }
            ]
        }
    ],
    "dataLoadTriggered": false,
    "state": {
        "value": "INGESTING_DATA"
    }
},
{
    "key": "api_1",
    "dataLoadTriggered": true,
    "state": {
        "value": "COMPLETED_SUCCESSFULLY"
    }
}
]
```

or standalone after creation or update, for example, cancelation.

In case of data upload

```
{
    "key": "api_1",
    "dataUploadTargets": [
        {
            "key": "prq_some_namespac_38",
            "name": "example_table_a",
            "namespace": "some_namespace",
            "fullyQualifiedName": "some_namespace.example_table_a",
            "persistenceMode": "APPEND",
            "columns": [
                {
                    "dataType": "DOUBLE",
                    "name": "CATEGORY"
                },
                {
                    "dataType": "STRING",
                    "name": "CATEGORY_NAME"
                },
                {
                    "dataType": "FORMATTED_TIMESTAMP",
                    "name": "CREATED",
                    "format": "yyyy/MM/dd HH:mm:ss"
                },
                {
```

```
                "dataType": "STRING",
                "name": "PROCESSOR"
            },
            {
                "dataType": "STRING",
                "name": "PROCESSOR_GROUP"
            }
        ]
    }
  ],
  "dataLoadTriggered": false,
  "state": {
    "value": "INGESTING_DATA"
  }
}
```

In case of data load

```
{
  "key": "api_1",
  "dataLoadTriggered": true,
  "state": {
    "value": "INGESTING_DATA"
  }
}
```

DATAINGESTIONCYCLESTATE

Used only as output, standalone or as part of a data ingestion cycle (DataIngestionCycle) (see above). Possible states are: ACCEPTING_DATA, INGESTING_DATA, COMPLETED_SUCCESSFULLY, CANCELED, and FAILED.

In case of 'FAILED', it will return a cause. Causes consist of a code and a message to indicate what happened exactly. The code is four digits long and always prefixed with "IER" for "Ingestion - Error".

```
{
  "value": "FAILED",
  "cause": {
    "code": "IER1000",
    "message": "An unexpected error occurred"
  }
}
```

TABLEDATA

Used only as input for data upload. Values must conform to the schema of the targeted source table. Null is a valid value.

```
[
    [1,"A","2021/05/10 12:13:14",1.1,"Distribution Center
Team","Distribution"],
```

```
    [2,"B","2021/06/11 15:16:17",2.2,"Distribution Center
Team","Distribution"],
    [3,"C","2021/07/12 18:19:20",3.3,null,"Sales"],
    [4,"D","2021/08/13 21:22:23",4.4,"Dealer Sales","Sales"]
]
```

STRINGCOLUMN

Used only as part of a source table definition (SourceTableDefinition) (see above).

LONGCOLUMN

Used only as part of a source table definition (SourceTableDefinition) (see above).

DOUBLECOLUMN

Used only as part of a source table definition (SourceTableDefinition) (see above).

FORMATTEDTIMESTAMPCOLUMN

Used only as part of a source table definition (SourceTableDefinition) (see above).

DEFAULTRESULT

Used only as standalone output, either when the operation performed does not have a dedicated result object itself (deletion of a source table, upload of source data) or when an error occurs on the server (any operation). The successful property of this object is either set to true or to false accordingly.

In case of false, the object will also contain a cause with a message.

```
{
  "successful": false,
  "cause": {
    "message": "An unexpected error occurred"
  }
}
```

APIVERSION

Used only as output after an API version check.

```
{
  "apiVersion": "3.2"
}
```

1.5 Endpoints

The following endpoints are available for your HTTP requests.

PATH SECTION: DATA SET

With the exception of the API version endpoint, all endpoints are used in the context of a specific data set. For this reason, all endpoints contain the following URL section:

/dataSets/{dataSet}

The {dataSet} parameter refers to the technical key of the data set and is replaced with the correct value at runtime. The value can be taken from the URL in the address bar of the browser when opening the corresponding data set.

The URL has the following form:

https://<host name>/#<project room>/dataCollection/y.dataset.<key>

The <key> parameter is based on the selected display name of the data set and should be readable. Use this key in all your API requests to this specific data set.

Example

https://ariscloud.com/#myprojectroom/dataCollection/y.dataset.mydataset

RETURN API VERSION

GET /api/pub/dataIngestion/version

Output: Current API version

LIST EXISTING SOURCE TABLE DEFINITIONS

GET /api/pub/dataIngestion/v1/dataSets/{dataSet}/sourceTableDefinitions

Input: Query parameter **'fqns'** to filter by fully-qualified names

Output: List of **SourceTableDefinition** objects

CREATE OR REPLACE SOURCE TABLES

POST /api/pub/dataIngestion/v1/dataSets/{dataSet}/sourceTables

Input:

- List of **SourceTableDefinition** objects
  - When creating a source table, a name, namespace, and columns are mandatory. Other properties are optional.
  - When replacing a source table,

    - an identifier of the existing table must be provided, either in the form of a key, fully-qualified name, or name and namespace. If more than one identifier is provided, the priority order is key > fully-qualified name > name and namespace. Identifiers of a lower priority are ignored if an identifier of a higher priority is present.

- Properties other than identifiers are optional. If not set, the values of the existing table are re-used. Note that the columns of a table are set as a whole, that means, to remove a column just ommit it from the body; to add a column repeat the information of the existing columns and include the new column.

- Query parameter **'forceReplace'** to signify that existing source tables with the same identifier should be replaced. In such a case, all previously stored source data is deleted. If not set to true, requests that would require replacement (for example, include an identifier of an existing table) are rejected.

Output:

List of **SourceTableDefinition** objects based on newly created or replaced source tables

RETURN IF DATA SET IS READY FOR INGESTION

POST /api/pub/dataIngestion/v1/dataSets/{dataSet}/readyForIngestion

Input: DataIngestionCycle containing

- either SourceTableDefinitions based on existing, fully-configured data sources to be updated. Any identifier can be provided. Other properties are optional and will be ignored.
- or a boolean flag to indicate that a data load should be started. Having a list of source table definitions and the value true for the mentioned flag is currently (2021Oct) not supported. If an upload followed by a data load is required, they have to be performed separately as two data ingestion cycles.

Output: IngestionReadyState. Please note that the readiness check for a data load will not necessarily consider all existing validation issues. So even if there are some validation issues displayed on the UI, it can be that the readiness check will return a positive result.

PREPARE FOR DATA INGESTION OR START DATA LOAD

POST /api/pub/dataIngestion/v1/dataSets/{dataSet}/ingestionCycles

Input: DataIngestionCycle containing

- either SourceTableDefinitions based on existing, fully-configured data sources to be updated. Any identifier can be provided. Other properties are optional and will be ignored.
- or a boolean flag to indicate that a data load should be started. Having a list of source table definitions and the value true for the mentioned flag is currently (2021Oct) not supported. If an upload followed by a data load is required, they have to be performed separately as two data ingestion cycles.

Ouput: New DataIngestionCycle

CANCEL DATA INGESTION

PUT
/api/pub/dataIngestion/v1/dataSets/{dataSet}/ingestionCycles/{ingestionCycle}/canceled

Output: Canceled DataIngestionCycle

UPLOAD SOURCE DATA

POST /api/pub/dataIngestion/v1/dataSets/{dataSet}/sourceTables/{sourceTable}/data

Input:

- Source table identifier as path parameter. Can either be a key or a fully-qualified name.
- List of objects as body, representing new source data entries.
  - The order of the columns corresponds to the order given upon creation of the source table and as returned by GET on the sourceTableDefinitions
  - Timestamp data can only be passed-in as strings formatted according to the date and time format of the corresponding source table column.
  - Bigger sets of data can be uploaded over multiple requests. Each request will lead to the data being stored in temporary form on the server side.

Output: Success result if data was received without error.

DELETE SOURCE TABLE DEFINITION AND DATA

DELETE /api/pub/dataIngestion/v1/dataSets/{dataSet}/sourceTables/{sourceTable}

Input: Source table identifier as path parameter. Can either be a key or a fully-qualified name.

Output: Success result if deletion was performed without error.

START DATA INGESTION

Indicate that all data has been uploaded and uploaded data can be ingested (persisted into the source database).

PUT /api/pub/dataIngestion/v1/dataSets/{dataSet}/ingestionCycles/{ingestionCycle}/dataComplete

Output: Running DataIngestionCycle

LIST EXISTING INGESTION CYCLES

GET /api/pub/dataIngestion/v1/dataSets/{dataSet}/ingestionCycles

Output: List of DataIngestionCycle objects

RETURN INGESTION CYCLE STATE

GET /api/pub/dataIngestion/v1/dataSets/{dataSet}/ingestionCycles/{ingestionCycle}/state

Output: DataIngestionCycleState

1.6 Persistence mode

All source tables have a persistence mode that determines how new data is processed on the server.

There are two different persistence modes:

OVERWRITE

This is the default mode and the setting to ensure the standard behavior from previous versions. If this mode is set, newly uploaded data overwrites the existing data. Data overwritten in this way is lost and cannot be recovered. If required, the overwritten data must be uploaded again.

APPEND

Instead of overwriting the already persisted table data on the server, newly uploaded data can also be appended by setting this mode. The new data rows are added at the end in the order in which they are received. This causes the size of the existing source table to grow. In a subsequent transformation and/or data load, the old and new data are then processed together. Persisting old data a second time with this setting (as if they were new rows) will lead to duplicate entries. This can have an impact on the correctness of analysis results.

Note that currently the only way to select a mode other than OVERWRITE for a source table is to use the Data Ingestion API to either create a new table or replace an existing one.

2 Support and legal information

This section provides you with some general information regarding product support and legal aspects.

2.1 Supported Web browsers

The following Web browsers are currently supported by ARIS Process Mining.

Browser	Version
Internet Explorer 11	Not supported
Microsoft Edge	Chromium-based version 79 or higher
Firefox	Firefox 74 or higher
Google Chrome	Google Chrome 80 or higher
Safari	Safari 11 or higher

The support for browsers of mobile devices will be available soon.

2.2 Documentation scope

The information provided describes the settings and features as they were at the time of publishing. Since documentation and software are subject to different production cycles, the description of settings and features may differ from actual settings and features. Information about discrepancies is provided in the Release Notes that accompany the product. Please read the Release Notes and take the information into account when installing, setting up, and using the product.

If you want to install technical and/or business system functions without using the consulting services provided by Software AG, you require extensive knowledge of the system to be installed, its intended purpose, the target systems, and their various dependencies. Due to the number of platforms and interdependent hardware and software configurations, we can describe only specific installations. It is not possible to document all settings and dependencies.

When you combine various technologies, please observe the manufacturers' instructions, particularly announcements concerning releases on their Internet pages. We cannot

guarantee proper functioning and installation of approved third-party systems and do not support them. Always follow the instructions provided in the installation manuals of the relevant manufacturers. If you experience difficulties, please contact the relevant manufacturer.

If you need help installing third-party systems, contact your local Software AG sales organization. Please note that this type of manufacturer-specific or customer-specific customization is not covered by the standard Software AG software maintenance agreement and can be performed only on special request and agreement.

2.3 Data protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR).

Where applicable, appropriate steps are documented in the respective administration documentation.

2.4 Restrictions

ARIS products are intended and developed for use by persons. Automated processes, such as the generation of content and the import of objects/artifacts via interfaces, can lead to an outsized amount of data, and their execution may exceed processing capacities and physical limits. For example, processing capacities are exceeded if an extremely high number of processing operations is started simultaneously. Physical limits may be exceeded if the memory available is not sufficient for the execution of operations or the storage of data.

Proper operation of ARIS products requires the availability of a reliable and fast network connection. Networks with insufficient response time will reduce system performance and may cause timeouts.

2.5 Support

If you have any questions on specific installations that you cannot perform yourself, contact your local Software AG sales organization (https://www.softwareag.com/corporate/company/global/offices/default.html). To get detailed information and support, use our websites.

If you have a valid support contract, you can contact **Global Support ARIS** at: **+800 ARISHELP**. If this number is not supported by your telephone provider, please refer to our Global Support Contact Directory.

ARIS COMMUNITY

Find information, expert articles, issue resolution, videos, and communication with other ARIS users. If you do not yet have an account, register at ARIS Community**.**

SOFTWARE AG EMPOWER PORTAL

You can find documentation on the Software AG Documentation website (https://empower.softwareag.com/). The site requires credentials for Software AG's Product Support site **Empower**. If you do not yet have an account for **Empower**, send an e-mail to empower@softwareag.com with your name, company, and company e-mail address and request an account.

If you have no account, you can use numerous links on the **TECHcommunity** website. For any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory and give us a call.

TECHCOMMUNITY

On the **TECHcommunity** website, you can find documentation and other technical information:

- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Access articles, code samples, demos, and tutorials.
- Find links to external websites that discuss open standards and web technology.
- Access product documentation, if you have **TECHcommunity** credentials. If you do not, you will need to register and specify **Documentation** as an area of interest.

EMPOWER (LOGIN REQUIRED)

If you have an account for **Empower**, use the following sites to find detailed information or get support:

- You can find product information on the Software AG Empower Product Support website.
- To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the Knowledge Center.
- Once you have an account, you can open Support Incidents online via the eService section of Empower.
- To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

SOFTWARE AG MANAGED LEARNINGS

Get more information and trainings to learn from your laptop computer, tablet or smartphone. Get the knowledge you need to succeed and make each and every project a success with expert training from Software AG.

If you do not have an account, register as a customer or as a partner.