



# Data Engineering Guide

Version 2.9  
2025-03-05

*Data Engineering Guide*

PDF generated on 2025-03-05

InterSystems® Data Fabric Studio™ Version 2.9

Copyright © 2025 InterSystems Corporation

All rights reserved.

InterSystems®, HealthShare Care Community®, HealthShare Unified Care Record®, IntegratedML®, InterSystems Caché®, InterSystems Ensemble®, InterSystems HealthShare®, InterSystems IRIS®, and TrakCare are registered trademarks of InterSystems Corporation. HealthShare® CMS Solution Pack™, HealthShare® Health Connect Cloud™, InterSystems® Data Fabric Studio™, InterSystems IRIS for Health™, InterSystems Supply Chain Orchestrator™, and InterSystems TotalView™ For Asset Management are trademarks of InterSystems Corporation. TrakCare is a registered trademark in Australia and the European Union.

All other brand or product names used herein are trademarks or registered trademarks of their respective companies or organizations.

This document contains trade secret and confidential information which is the property of InterSystems Corporation, One Congress Street, Boston, MA 02114, or its affiliates, and is furnished for the sole purpose of the operation and maintenance of the products of InterSystems Corporation. No part of this publication is to be used for any other purpose, and this publication is not to be reproduced, copied, disclosed, transmitted, stored in a retrieval system or translated into any human or computer language, in any form, by any means, in whole or in part, without the express prior written consent of InterSystems Corporation.

The copying, use and disposition of this document and the software programs described herein is prohibited except to the limited extent set forth in the standard software license agreement(s) of InterSystems Corporation covering such programs and related documentation. InterSystems Corporation makes no representations and warranties concerning such software programs other than those set forth in such standard software license agreement(s). In addition, the liability of InterSystems Corporation for any losses or damages relating to or arising out of the use of such software programs is limited in the manner set forth in such standard software license agreement(s).

THE FOREGOING IS A GENERAL SUMMARY OF THE RESTRICTIONS AND LIMITATIONS IMPOSED BY INTERSYSTEMS CORPORATION ON THE USE OF, AND LIABILITY ARISING FROM, ITS COMPUTER SOFTWARE. FOR COMPLETE INFORMATION REFERENCE SHOULD BE MADE TO THE STANDARD SOFTWARE LICENSE AGREEMENT(S) OF INTERSYSTEMS CORPORATION, COPIES OF WHICH WILL BE MADE AVAILABLE UPON REQUEST.

InterSystems Corporation disclaims responsibility for errors which may appear in this document, and it reserves the right, in its sole discretion and without notice, to make substitutions and modifications in the products and practices described in this document.

For Support questions about any InterSystems products, contact:

**InterSystems Worldwide Response Center (WRC)**

Tel: +1-617-621-0700

Tel: +44 (0) 844 854 2917

Email: [support@InterSystems.com](mailto:support@InterSystems.com)

# Table of Contents

<b>1 Welcome, Data Engineers (2.9)</b>	<b>1</b>
1.1 Data Sources and Schemas	1
1.2 Recipes	1
1.3 Analytics Cubes	2
1.4 Snapshots	2
1.5 Scheduling	2
1.6 See Also	3
<b>2 Importing Schemas (2.9)</b>	<b>5</b>
2.1 Adding Tables from a JDBC Data Source	5
2.2 Adding an SQL Query from a JDBC Data Source	6
2.3 Adding a File-Based Schema	6
2.4 Filter Options When Importing Schemas	7
2.5 See Also	7
<b>3 Schema Evolution (2.9)</b>	<b>9</b>
3.1 How the System Manages Changes to Schemas	9
3.2 See Also	9
<b>4 Editing and Managing Schemas (2.9)</b>	<b>11</b>
4.1 Viewing the Data Catalog	11
4.2 Displaying a Schema	11
4.3 Editing a Schema	12
4.4 Schema Options	12
4.5 Editing Schema Fields	15
4.6 Publishing a Schema	15
4.7 Reimporting a Schema	15
4.8 Deleting a Schema	16
4.9 See Also	16
<b>5 Streaming Data (2.9)</b>	<b>17</b>
5.1 Configuring a Broker	17
5.2 Configuring a Schema Registry	18
5.3 Testing a Broker	19
5.4 Configuring a Topic	19
<b>6 Defining and Managing Categories (2.9)</b>	<b>21</b>
6.1 Defining a Category	21
6.2 Renaming a Category	21
6.3 Deleting a Category	22
6.4 See Also	22
<b>7 Introduction to Recipes and Staging Tables (2.9)</b>	<b>23</b>
7.1 Structure of a Recipe	23
7.2 Staging Tables	24
7.3 See Also	24
<b>8 Defining Recipes (2.9)</b>	<b>25</b>
8.1 Creating a Recipe	25
8.2 Adding a Staging Activity	25
8.3 Adding a Transformation Activity	27

8.4 Adding a Validation Activity .....	27
8.5 Adding a Reconciliation Activity .....	28
8.6 Viewing the Staging Table Details .....	30
8.7 Adding a Promotion Activity .....	30
8.8 Adding a Promotion Activity Item .....	31
8.8.1 Specifying the Target File Settings .....	32
8.9 Adding a Custom Activity .....	32
8.10 Step Mode .....	33
8.11 See Also .....	33
<b>Activity Type Reference (2.9) .....</b>	<b>35</b>
Staging Activity (2.9) .....	36
Transformation Activity (2.9) .....	37
Validation Activity (2.9) .....	39
Reconciliation Activity (2.9) .....	42
Promotion Activity (2.9) .....	43
Custom Activity (2.9) .....	44
<b>9 Editing and Managing Recipes (2.9) .....</b>	<b>45</b>
9.1 Recipe Changes .....	45
9.2 Viewing the Recipes .....	45
9.3 Editing a Recipe .....	46
9.4 Publishing an Activity .....	47
9.5 Disabling an Activity .....	47
9.6 Enabling an Activity .....	47
9.7 Deleting an Activity .....	48
9.8 Renaming a Recipe .....	48
9.9 Moving a Recipe Into or Out of a Group .....	48
9.10 Copying a Recipe .....	49
9.11 Disabling a Recipe .....	49
9.12 Enabling a Recipe .....	49
9.13 Purging Old Recipe Data .....	50
9.14 Deleting a Recipe .....	50
9.15 Recovering a Recipe .....	50
9.16 Permanently Deleting a Recipe .....	51
9.17 See Also .....	51
<b>10 Managing Recipe Groups (2.9) .....</b>	<b>53</b>
10.1 Creating a Recipe Group .....	53
10.2 Renaming a Recipe Group .....	53
10.3 Deleting a Recipe Group .....	54
10.4 Recovering a Deleted Recipe Group .....	54
10.5 See Also .....	54
<b>11 Defining Snapshots (2.9) .....</b>	<b>55</b>
11.1 Uses of Snapshots .....	55
11.2 Snapshot Tables .....	55
11.3 Defining a Snapshot .....	55
11.4 See Also .....	57
<b>12 Editing and Managing Snapshots (2.9) .....</b>	<b>59</b>
12.1 Viewing the Snapshots Dashboard .....	59
12.2 Editing a Snapshot .....	59

12.3 Retagging a Snapshot Run .....	60
12.4 Disabling a Snapshot .....	60
12.5 Enabling a Snapshot .....	60
12.6 Deleting a Snapshot .....	61
12.7 See Also .....	61
<b>13 Managing Table Indices (2.9) .....</b>	<b>63</b>
13.1 Adding an Index to a Staging Table .....	63
13.2 Adding an Index to a Snapshot .....	63
13.3 Index Build Pending Message .....	64
13.4 See Also .....	64
<b>14 Defining Entities and Calendars (2.9) .....</b>	<b>65</b>
14.1 Defining an Entity .....	65
14.2 Defining a Child Entity .....	66
14.3 Editing an Entity .....	66
14.4 Editing a Holiday Calendar .....	67
14.5 Deleting an Entity .....	68
14.6 See Also .....	68
<b>15 Scheduling and Running Tasks (2.9) .....</b>	<b>69</b>
15.1 Scheduling a Task .....	69
15.2 Scheduling Details .....	71
15.3 Managing Task Dependencies .....	71
15.4 Functions Available in Dependency Expressions .....	72
15.5 Modifying a Task .....	73
15.6 Running a Task Manually .....	73
15.7 Aborting a Task .....	73
15.8 Deleting a Task .....	74
15.9 See Also .....	74
<b>16 Managing Task Groups (2.9) .....</b>	<b>75</b>
16.1 Creating an Ordinary Group .....	75
16.2 Creating a Schedule Group .....	75
16.3 Moving a Task into a Group .....	76
16.4 Moving a Task out of a Group .....	76
16.5 Renaming a Task Group .....	76
16.6 Deleting a Task Group .....	77
16.7 See Also .....	77
<b>17 Handling Task Errors (Using the Workflow Inbox) (2.9) .....</b>	<b>79</b>
17.1 When the System Encounters an Error .....	79
17.2 First Steps .....	79
17.3 Viewing the Workflow Inbox .....	80
17.4 Addressing a Task .....	80
17.5 Relinquishing a Task .....	81
17.6 See Also .....	81
<b>18 Viewing Run History (2.9) .....</b>	<b>83</b>
18.1 Viewing Run History .....	83
18.2 Task Status .....	84
18.3 Downloadable Run Reports (Recipes) .....	84
18.4 See Also .....	85
<b>19 Filtering and Customizing the Business Scheduler (2.9) .....</b>	<b>87</b>

19.1 Filtering the Display .....	87
19.2 Customizing the Columns .....	87
19.3 See Also .....	87
<b>20 Using the File Manager (2.9) .....</b>	<b>89</b>
20.1 Standard Subdirectories for a File Source .....	89
20.2 Viewing the Directory Contents for a File Source .....	89
20.3 Uploading Files .....	90
20.4 Deleting a File .....	91
20.5 See Also .....	91
<b>21 Exporting and Importing Configurations (2.9) .....</b>	<b>93</b>
21.1 Exporting Configurations .....	93
21.2 Editing Configurations .....	93
21.3 Importing Configurations .....	94
21.4 See Also .....	94

# 1

## Welcome, Data Engineers (2.9)

As a data engineer for your [InterSystems Data Fabric Studio™](#) solution, your task is to set up the *data pipeline*, a term that refers to the infrastructure needed to make data available within the solution (or to push the data downstream). This page provides an overview of that infrastructure, from start to finish.

When data is available within the system, you and [data analysts](#) can create analysis cubes and reports based on those tables.

### 1.1 Data Sources and Schemas

In the product, a *data source* is a named configuration item that provides all the information needed to retrieve data from an external source. A classic data source is a database, but another possibility is an external system that provides an API via which an authenticated user can retrieve data. Yet another option is delimited files, either retrieved from a cloud location or pushed to the local file system. In all cases, the [data source](#) definition, usually configured by an administrator, contains all the information needed to retrieve structural information as well as data.

A data source can provide multiple kinds of data, each with a specific structure. The classic case is a database that contains many tables, where each table has a specific structure. Similarly, a specific API call returns data in a specific format. In the product, each of these structures is a *schema*, and the goal is to create a Data Catalog that consists of the schemas that are relevant to the business (and that are ultimately needed for reports and analysis). As a data engineer, your task is to import schemas from the data sources into the Data Catalog and then refine them and categorize them in ways that are useful to your organization.

To get started defining schemas, see [Importing Schemas](#).

### 1.2 Recipes

*Recipes* describe how to load data from external sources into Data Fabric Studio or into external tables or files. Any recipe consists of some or all of the following steps, in order:

1. Staging activities, each of which loads data into a staging table.
2. Transformation activities, which can clean data in various ways.

Transformation activities update the staging table, either by adding new fields or overwriting existing fields. You can directly examine the staging table at any time.

3. Validation activities, which can compare data to desired ranges, as an example.

4. Reconciliation activities, in which you specify comparisons that define a valid reconciliation of the data.

Each validation and reconciliation activity writes a report file with any errors. For either kind of activity, an error can either halt processing or simply result in a warning message, as you choose. In all cases, an appropriate user role is alerted via the [workflow](#) module so that the data can be examined and corrected if needed, and the recipe can be rerun.

5. Data promotion activities, which use SQL to extract data from the staging tables and update final tables or external files. In the case of final tables, those can be in the native database or can be in an external database.

A recipe can also include custom steps at any stage in the processing.

To get started with this task, see [Defining Recipes](#).

## 1.3 Analytics Cubes

Data Fabric Studio includes InterSystems IRIS® Adaptive Analytics, a Business Intelligence tool powered by co-development with [AtScale](#). This means that data engineers and [data analysts](#) can define cubes based on those tables, and then use those cubes for analytics. See [Defining Cubes](#).

## 1.4 Snapshots

Data Fabric Studio provides an additional mechanism to support your data needs: snapshots. With snapshots, you can easily save data for later inspection by regulators; the snapshot can pull data from multiple tables as needed, writing them to a snapshot table, and the system applies a tag to the records. The product automatically stores all snapshot runs; that is, a new snapshot run does not overwrite a previous snapshot run.

When you have run a snapshot multiple times, applying a different tag each time, you can examine how that data changes over time. In particular, you can build a cube on the snapshot data, using the tag values as a dimension.

See [Defining Snapshots](#).

## 1.5 Scheduling

The Business Scheduler provides an easy way to schedule the running of tasks: running recipes, building cubes, and performing snapshots. See [Scheduling and Running Tasks](#).

In the initial phases of implementation, you can configure tasks to be run manually (from the Business Scheduler).

Later, when you want the tasks to be executed on a schedule, it is necessary to define the applicable calendar information and to manage dependencies among tasks. This works as follows:

- To define calendar information, you define *entities*, each which has its own calendar. To simplify scheduling, the product supports a hierarchical system of entities, each of which can have its own business calendar but can inherit calendar details from its parent. See [Defining Entities and Calendars](#).
- To specify dependencies among tasks, you apply a tag to a scheduled task and define dependency expressions in other tasks, referring to that tag. Then a task can be run only after the dependencies are fulfilled. See [Managing Task Dependencies](#).



## 1.6 See Also

- [Welcome, Data Analysts](#)
- [About Your Solution: What Is Not Documented](#)



# 2


## Importing Schemas (2.9)

This page describes how to import the *schemas* that make up the [Data Catalog](#). Each of these schemas (not to be confused with SQL schemas) is associated with a specific external [data source](#) and describes the structure of a single data element such as a table or a delimited file provided by that data source.


After importing a schema, it is generally necessary to [edit](#) the details such as the data extraction strategy and field types.

### 2.1 Adding Tables from a JDBC Data Source


To add one or more tables from a JDBC data source:

1. Click the Data Catalog  icon in the application menu.
2. Click **Data Schema Importer**.
3. Select a JDBC data source from the **Data Source** dropdown menu.
4. Select an SQL schema (group of tables) from the **Schema** dropdown list.

The section **Available Items from JDBC Source** then displays the tables in the selected SQL schema.

5. Add tables to be imported in either of two ways:
  - Click the check box next to each table name and then click **Select for Import**.
  - Click the Add to Imports  button in the applicable row or rows.


The table or tables are then moved to the section **Items Selected For Import**; the system has not yet imported their metadata.

6. Optionally select a different item from the **Schema** dropdown list and repeat these steps as needed. This process adds to the list in **Items Selected For Import**; the system has not yet imported their metadata.
7. Optionally, to remove a table from the list in **Items Selected For Import**, click the Delete  icon in the applicable row.
8. To import the metadata for the selected tables, click **Import**.

The system then imports metadata for all the selected tables and displays the **Results of Last Import** tab, which lists all the tables whose metadata it just imported.

## 2.2 Adding an SQL Query from a JDBC Data Source

Instead of (or in addition to) adding [tables](#) from a JDBC data source, you can directly use a custom SQL SELECT query. To do so:

1. Click the Data Catalog  icon in the application menu.
2. Click **Data Schema Importer**.
3. Select a JDBC data source from the **Data Source** dropdown menu.
4. Click the **SQL Query** tab.
5. Type an SQL SELECT statement into the box.
6. Optionally modify **Row Count**, which controls the number of rows that are sampled and displayed.
7. Click **Run Query**.

The page then displays the query results, which you can use to verify that the query is as expected.

8. To add a schema to the catalog based on this query, click **Save to Catalog**.

The system then displays a dialog box.

9. For **Name**, type a short, unique name for the new schema. You cannot change this name later.
10. For **Description**, type an optional description of the new schema.

The system then generates metadata for the query and displays the new schema, which you can now edit as described in [Editing Schemas](#).

## 2.3 Adding a File-Based Schema

A file-based data source is a UTF-8 encoded file with one record per line, where each line follows a convention that uses a specific delimiter between fields (typically a comma or a tab). This convention implicitly defines a schema. To define this schema within the product, you need to upload and then import a sample file and specify the delimiter and the field names. (Excel files are a special case of a file-based data source in which you do not need to specify the delimiter that separates fields.)

[FileDir](#), [ExcelSingleFileDir](#), and [S3Delimited](#) data sources are all file-based data sources.


To add a file-based schema:

1. Obtain a sample file for the schema, applicable to a specific file-based data source.

The sample file can consist of only one line. Also, the sample file must have a header row—an initial line that contains the names of the fields. (In other words, the sample file may or may not contain any actual data.)

2. Rename the sample file so it has an appropriate and useful short filename, because this short filename becomes the schema name within the system (with punctuation characters removed), and it cannot be edited.

For example, if you load a sample file named `sampledata.csv`, that becomes a schema named `sampledatacsv`.


3. Use the [File Manager](#) to upload this file to the `Samples` directory used by the chosen file-based data source.
4. Click the Data Catalog  icon in the application menu.

5. Click **Data Schema Importer**.

6. Select the chosen file-based data source.


The section **Available Items from Source** lists the files in the Samples [subdirectory](#) for this data source.

7. Add files to be imported in either of two ways:

- Click the check box next to each filename and then click **Select for Import**.
- Click the Add to Imports  button in the applicable row or rows.

The filename or filenames are then moved to the section **Items Selected For Import**; the system has not yet imported their metadata.

8. If the right side of the page contains a long list of files to import, you may want to double check the list. In doing so, if you want to filter this display, you can type into the **Members Selected for Import** filter. This option affects what this page *displays*, but does not affect the import of metadata.

9. Optionally, to remove a file from this list, click the Delete  icon in the applicable row.

10. To import the selected files, click **Import**.

The system then imports all the selected files and generates metadata from them.

The page then displays the **Results of Last Import** tab, which lists all the files it imported.

## 2.4 Filter Options When Importing Schemas

Depending on the type of data source, there may be a large number of schemas to choose among. The import page provides options to help you filter the display. You can filter and sort the items listed in the **Available Items From Source** section on the left, as well as the items listed in the **Items Selected For Import** on the right. In both areas, you can do the following:

- To show only names containing a specific string, type that string into the filter above the **Item Name** column.
- To display items based on whether they are in the catalog, click the dropdown above the **In Catalog** column and select either or both of the following check boxes as needed:
  - **Yes**—display the items that are currently in the catalog.
  - **No**—display the items that are not currently in catalog.
- You can click the **Item Name** and **Item Name** column headers to sort those columns.

**Important:** These filters do not affect what is imported when you click **Import**. The only change is to the display of the items on this page.

## 2.5 See Also

- [Defining Data Sources](#)
- [Using the File Manager](#)
- [Editing and Managing Schemas](#)

- [Defining Categories](#)

# 3

## Schema Evolution (2.9)

This page explains how InterSystems Data Fabric Studio™ accommodates schema changes while also preserving your data, a process known as *schema evolution*.

### 3.1 How the System Manages Changes to Schemas

Schema evolution works as follows:

1. When you [import](#) a schema and first edit it, the schema is a draft and cannot be used by recipes.
2. When you [publish](#) a schema, it can then be used by recipes—specifically in a staging activity (which loads data into the system).
3. When you define a [recipe](#), each of its activities is initially a draft.
4. When you publish a staging activity of a recipe, the system then generates the table that will store the data to be loaded.
5. When you [edit](#) or [reimport](#) a schema, you are implicitly creating a draft that is not yet used by anything. The existing schema is unchanged; existing data is also unchanged.

While a schema is in draft state, you have the option of deleting the draft (and reverting to the previous published version).

6. When you [republish a schema](#), if the change affects the structure needed to contain the data, the system automatically increments the version number of the schema and automatically generates a new draft of any staging activity that is affected by this change. The new staging activity draft is not automatically published, and the previous staging activity is unchanged.
7. When you [republish a staging activity](#), the system generates a new table as needed, considering the structural change that has occurred. This leaves the old table (and its data) unchanged.

Similarly, changes to earlier parts of a recipe can affect later activities in a recipe, requiring changes in those later activities. The system detects these changes, automatically generates draft activities where appropriate, and notifies you of the drafts.

### 3.2 See Also

- [Editing and Managing Schemas](#)
- [Recipes and Staging Tables](#)

- [Defining Recipes](#)



# 4

## Editing and Managing Schemas (2.9)

In InterSystems Data Fabric Studio™, a schema describes the structure of a single data element such as a table or a file using a specific file format. This page describes how to edit and manage schemas; another page provides information on [importing](#) them.

### 4.1 Viewing the Data Catalog

To view the Data Catalog and the schemas currently contained in it:

1. Click the Data Catalog  icon in the application menu.
2. Click **Data Catalog Browser**.

This page enables you to see all available schemas, either as a table or as a set of rectangular cards.


For each schema, the page displays the name of the data source that contains this schema, the actual name of the schemas, and any [categories](#) to which it belongs. If you click **Card View**, the cards also display the descriptions.

3. Optionally filter the display by using the options at the top of the page:
  - **Schema Definition Item Name**—Filters the display to include only the schemas whose short names contain the given string.
  - **Data Source**—Filters the display to include only schemas from a specific [data source](#).
  - **Category**—Filters the display to include only schemas labelled with a specific [category](#). Note that schemas can be labelled with any number of categories, including no categories at all.
  - **Recipe**—Filters the display to include only schemas used in a specific [recipe](#).
  - **Drafts** —If selected, displays the schemas that are [drafts](#). If cleared, these schemas are not displayed.
  - **Reset** —Clears all the filter options.



Click any schema in order to view it or make changes.

### 4.2 Displaying a Schema

To display a schema:



1. Click the Data Catalog  icon in the application menu.
2. Click **Data Catalog Browser**.
3. Click the schema name.

If there is a draft of the schema, you can choose to display the published version or display the draft (which is shown as a separate item as in the following example).

Data Source	Item Name
 filedir	sampledatacsv
 filedir	comparisondatacsv
DRAFT  filedir	comparisondatacsv

## 4.3 Editing a Schema

To edit a schema:

1. First [display](#) it.
2. If the schema has been published, click **Edit Schema** in the upper right to make the schema editable. (If there is a draft of the schema and you are viewing the published version, this button opens the draft version of the schema.)
3. Then make any of the following changes:
  - To add or edit a description, click the Edit  icon next to **Description** and then make edits.
  - To add a category, click **Add Category...** and then click a category.
  - To remove a category, click the X icon for the category.
  - To modify the details of how the data source is handled, click the Edit  icon in that right box. This displays a dialog box where you can make changes. The details depend on the kind of schema; see [Schema Details](#).

When you are done, click **Save** to save these changes or **Cancel** to discard them.

- To make changes to the schema fields, see [Editing Schema Fields](#).
4. Click **Save Draft** to save these changes.
- Or click **Delete Draft** to discard them (and return to the previous schema definition).

Note that if you leave this page without saving your changes, the system automatically saves your changes as a draft.

## 4.4 Schema Options

This section provides reference information on the options that control how a schema (and its data source) are interpreted and used.

### Extraction Strategy

*Applies to all schema types.* Specifies which records to use from the data source.

For SQL-based schemas (tables and custom queries), the options are as follows:

- **General Table**—Retrieves all new records and any changed records.  
*A new record* is a record with a new primary key value. *A changed record* is a record with an existing primary key value but a change to one or more other fields.
- **Append only transactional table with a numeric sequential key**—Retrieves all new records since the last sequential primary key.  
If you specify this, also specify **Extraction Strategy Field**, which must be the name of the field that contains the sequential primary key.
- **Transactional table with a changed timestamp field**—Retrieves all records changed since the last timestamp.  
If you specify this, also specify **Extraction Strategy Field**, which must be the name of the field that contains the timestamp.
- **Simple load**—Retrieves all records, whether or not the system has them already.
- **Change data capture**—Uses the Snowflake [Change Data Capture](#) process. This applies only to Snowflake data sources.

For file-based schemas, the options are **General Table** and **Simple Load**.

#### UsePrimaryKeyPseudocolumn

*Applies to table-based schemas.* Specifies whether to use a pseudocolumn in this table as the primary key. If you select this option, also specify **PrimaryKeyPseudocolumnName**, which must be the name of the pseudocolumn.  
*Does not apply to a schema based on a custom query.*

#### CreateForeignTable

*Applies to SQL-based schemas (tables and custom queries).* Projects this table or query as a foreign table, which is read-only and which does not copy the data into Data Fabric Studio.

If you select this option, you can override the default schema used for foreign tables from this data source. To do so, select **ForeignTableLocalSchemaOverride** and then specify a schema name for **ForeignTableLocalSchema**. (An additional override is available for individual tables, if needed.)

#### Delimiter

*Applies to file-based schemas.* Specifies the delimiter that separates fields in the file.

When loading delimited files, you must define and follow specific conventions so that they can be read appropriately. The first consideration is the delimiter that separates the fields, controlled by the **Delimiter** setting.

If the character specified by **Delimiter** is also used *within* any field, you must wrap the field in quotation marks (single or double, as you choose—or possibly some other character). To control which character is used to wrap a field, you specify the **QuoteChar** option, whose default is the double quote character ". If the character specified by **QuoteChar** is used *within* any field that is already wrapped in quotation marks, see the **DoubleQuote** and **EscapeChar** settings.

#### FileNamePattern

*Applies to file-based schemas.* Specifies which files are read and used as input, for this schema. That is, input files whose names conform to this pattern are processed, and other files are ignored. You can specify any legal filename string or a filename wildcard expression, consistent with the operating system. The initial pattern is simply the name of the file that you initially loaded.

**FileNamePatternMatching**

*Applies to file-based schemas.* Specifies the form of pattern matching to use (for **FileNamePattern**).

**CommentLinePrefix**

*Applies to file-based schemas.* Specifies the prefix that indicates the start of a comment line in this file. A comment line is ignored.

**DoubleQuote**

*Applies to file-based schemas.* Specifies how to interpret a pair of quotation marks contained within a field that is wrapped in quotation marks. (Here the term *quotation marks* refers to the character specified by the **QuoteChar** option.)

If this option is **Yes**, then in the stated context, a pair of quotation marks is interpreted as a single quotation mark.

For example, suppose that **Delimiter** is a comma, **QuoteChar** is `"`, and **DoubleQuote** as **Yes**. Then the value `"here is a "value" to load"` is loaded as `here is a "value" to load`

**EscapeChar**

*Applies to file-based schemas.* Specifies the character used to start an escape sequence, so that you can include a quotation mark within a field that is wrapped in quotation marks. *Applies to file-based schemas.* Specifies how to interpret a pair of quotation marks contained within a field that is wrapped in quotation marks. (Here the term *quotation marks* refers to the character specified by the **QuoteChar** option.)

For example, suppose that **QuoteChar** is `"` and **EscapeChar** is a backslash `\`. Then the value `"here is a \"value\" to load"` is loaded as `here is a "value" to load`

**LineTerminator**

*Applies to file-based schemas.* Specifies the character or sequence of characters that indicate the end of a line in this file.

**QuoteChar**

*Applies to file-based schemas.* Specifies the character used to wrap any field that contains the field delimiter character. For example, if fields are delimited by a comma, then if a field contains a comma, that field must be wrapped in a quotation mark, as specified by **QuoteChar**.

If the character specified by **QuoteChar** is used *within* any field that is already wrapped in quotation marks, see the **DoubleQuote** and **EscapeChar** settings.

**SkipInitialSpace**


*Applies to file-based schemas.* Specifies how to handle space characters that appear within a field after the field delimiter. If this option is **Yes**, initial space characters are trimmed when the data is loaded. Note that you can also define a [transformation](#) that trims whitespace on a field-by-field basis.

**Header Row**

*Applies to file-based schemas.* Specifies whether the files have a header row. Select **Yes** or **No**. The header row is ignored when the data is loaded.

## 4.5 Editing Schema Fields

For any kind of data source, when you edit the schema, the bottom part of the page displays a table labeled **Schema Fields**. Here you can modify the schema fields as follows:

- To specify the primary key for this schema, click the **Primary Key** check box for all the fields that make up the primary key.
- To modify other details for a specific field, click the Edit Attributes  icon for that field. This displays a dialog box where you can edit the following:
  - **Data Type**—Specifies the data type for this field.
  - **Description**—Specifies an optional description for the field.
  - **Required**—Specifies whether this field is permitted to be null. If a field is required, an error is thrown if the value is null.
  - **Default Value**—Specifies the value to use if this field is null.
  - **Data Format** (for date, time, and timestamp fields)—Select the format that data is expected to be in for this field.
  - **Min Value** (for double and integer fields)—Specifies the minimum allowed value of this field.
  - **Max Value** (for double and integer fields)—Specifies the maximum allowed value of this field.
  - **Scale Value** (for double fields)—Specifies the number of digits to display after the decimal point. This has no effect on how data is stored.
  - **Length Value** (for string fields)—Specifies the maximum number of characters allowed in this field.

When you are done with this dialog box, click **Save** to save these changes or **Cancel** to discard them.

## 4.6 Publishing a Schema

A schema cannot be used in [recipes](#) unless it has been published. To publish a schema:

1. [Display](#) it.
2. Click **Publish Schema** in the upper right.

## 4.7 Reimporting a Schema

To reimport a schema:

1. [Display](#) it.
2. Click **Re-import Schema** in the upper right. This change automatically creates a new draft of the schema.
3. [Edit](#) the schema again as needed.

If the schema has not yet been [published](#), you cannot reimport it. You can, however, delete it and then import it again.

## 4.8 Deleting a Schema

To delete a schema:

1. [Display](#) it.
2. Click **Delete** in the upper right.
3. Click **Delete** to confirm.

You cannot delete a schema that is used in a [recipe](#); if you attempt to do so, you are notified about any recipe activities that depend on this schema. You can then go edit the recipe to remove the dependency.

## 4.9 See Also

- [Importing Schemas](#)
- [Defining Categories](#)
- [Defining Recipes](#)


# 5

## Streaming Data (2.9)

You can configure the solution to receive streaming data, written directly into tables that you can use within recipes; this refers specifically to streaming data from the Apache Kafka streaming platform. The process starts with configuring and testing one or more [brokers](#), including credentials as needed for authentication. Once the connections are [tested](#), you configure one or more topics for each broker. Each topic represents a stream of data that follows a specific schema, and you specify the table into which this data is written. This table is then available as a schema within the [Data Catalog](#).

### 5.1 Configuring a Broker

To configure a broker:

1. Click the Streaming  icon in the application menu.  
The system then displays any existing brokers.
2. Click **New Broker**.
3. Enter values as follows:
  - **Broker Name** — A unique name for the broker, for use within this system.
  - **Broker Server URL**—Host and port to use when establishing the initial connection to the broker cluster. Use a string of the following form:  

```
host:port
```
  - **Broker's Security Protocol**—Select the option that describes how to connect to this broker. These protocols describe both whether the connection is encrypted as well as the form of authentication to use. Select one of the following and specify details as follows:
    - **Unauthenticated, unencrypted channel (PLAINTEXT)**—With this protocol, the connection is not encrypted and no credentials are used.
    - **SASL authenticated, unencrypted channel (SASL\_PLAINTEXT)**—With this protocol, the connection is not encrypted and SASL credentials are used.  
If you select this protocol, also specify **Basic Auth Credentials**; for this value, select an existing [credential](#).
    - **SASL authenticated, SSL channel (SAL\_SSL)**—With this protocol, the connection is encrypted and SASL credentials are used.  
If you select this protocol, also do the following:

- a. Click **Select files** and upload the key store (.jks) file that contains the private key to use when encrypting the channel.
  - b. For **SSL Trust Store's Password**, enter the password for the key store file.
  - c. For **Confirm SSL Trust Store's Password**, enter the same password.
  - d. For **Basic Auth Credentials**, specify the credentials to use; select an existing [credential](#).
- **SSL channel (SSL)**—With this protocol, the connection is encrypted and no credentials are used.
- If you select this protocol, also do the following:
- a. For **SSL Trust Store**, click **Select files** and upload the trust store file containing certificates from certificate authorities (CAs).
  - b. For **SSL Trust Store's Password**, enter the password for the trust store file, if needed.
  - c. For **SSL Key Store**, click **Select files** and upload the key store file containing a private key.
  - d. For **SSL Key Store's Password**, enter the password for the key store file, if needed.
  - e. For **SSL Key Password**, enter the password for the private key held in the key store file.
  - f. For **SSL Provider**, optionally enter the name of the security provider used for the SSL connection.
  - g. For **SSL Cipher Suites**, optionally enter a named combination of authentication, encryption, MAC and key exchange algorithm used to negotiate security settings.
  - h. For **SSL Enabled Protocols**, optionally enter a list of protocols enabled for the SSL connection.
  - i. For **SSL Trust Store Type**, optionally specify the file format of the trust store.
  - j. For **SSL Key Store Type**, optionally specify the file format of the key store.
- **Schema Registry URL** (optional) — See [Configuring a Schema Registry](#).

4. Click **Submit**.

## 5.2 Configuring a Schema Registry

When you configure a broker, you can configure an optional schema registry for it, which will simplify the process of defining topics. A [schema registry](#) provides data governance. It acts as a central location for managing and validating the schemas of topic message data. In addition, the schema registry offers serializers and deserializers for converting events to and from binary strings for transit.

To configure a schema registry for a broker, specify the following values while configuring or reconfiguring the broker:

- **Schema Registry URL** (optional) — Specify the full URL for the registry, including either `http:` or `https:`, depending on whether the connection is encrypted.
- **Schema Registry's Security Protocol** — Select the option that describes how to connect to the registry. These protocols describe both whether the connection is encrypted as well as the form of authentication to use. The choices are the same as for the connections to the broker; see the [previous section](#). The default is `SASL_SSL`.



## 5.3 Testing a Broker

To test the configuration for a broker, right click the broker and select **Test connection**. The system then tests the connection and displays a dialog box indicating whether the test was successful.

## 5.4 Configuring a Topic

To configure a topic:

1. Obtain a copy of the schema of the messages for this topic, in JSON format.

2. Click the Streaming  icon in the application menu.

The system then displays the brokers.

3. Click the broker with which the topic is associated.

4. Click **New Topic** in the upper right.

5. Specify the following details:

- **Topic Name**—Name of the topic (as defined in the broker).
- **Message Schema**—Schema of the messages for this topic, as obtained earlier.
- **Class Name**—Name of the InterSystems IRIS persistent class to create, to contain the data received for this topic. This class definition does not have to exist. Specify a fully qualified class name in the form *Package.Class* (or *Package.Package.Class* and so on). Note that any package name must either be new or must exactly match an existing package name, including case.
- **Polling Time**—Specifies the intervals of time (in milliseconds) at which the stream is polled before all collected messages are written to the database.
- **Notification Interval**—Specifies the intervals of time (in milliseconds) at which the streaming process collects statistics and checks for a stop signal.
- **Consumer Group**—ID for the consumer group that will process messages from this topic in parallel.
- **Message Key Deserializer**—Deserializer to use when converting the message key back into a readable object.
- **Message Value Deserializer**—Deserializer to use when converting the message back into a readable object.
- **Retrieve message schema from schema registry at runtime using subject and version**—Optionally select this check box if you want to retrieve the message schema from the schema registry at runtime.

If you select this option, also specify:

- **Schema Subject**—Name of the message's data model as stored in the schema registry. A schema is a particular version of a subject. The message schema does not need to be provided directly if the schema's subject name and version are supplied instead. The schema can then be accessed via the schema registry's API.
- **Schema Version**—Schema version to use.

6. Click **Submit**.




# 6

## Defining and Managing Categories (2.9)

*Categories* enable you to define groupings of schemas in the [Data Catalog](#); each schema can belong to any number of categories (including none at all). Then when you search for a schema, you can use the categories as filters.

### 6.1 Defining a Category

To create a category:

1. Click the Data Catalog  icon in the application menu.

2. Click **Categories**.

The system displays any existing categories.

3. Click **New Category** in the upper right.


4. For **Name**, enter a short, unique name.

Category names cannot differ only by case. For example, you cannot have a category named `sample` and another category named `SAMPLE`.

5. Click **Submit**.


### 6.2 Renaming a Category

To rename a category:

1. Click the Data Catalog  icon in the application menu.

2. Click **Categories**.

The system displays any existing categories.

3. Click the Edit  icon in the row for the category.



4. For **Name**, enter a new name.

5. Click **Submit**.

The category name is automatically updated in all places where it is used.

## 6.3 Deleting a Category

To delete a category:

1. Make sure that no schema belongs to this category. To do this, display the [Data Catalog](#), filter to this category, and edit the schemas as necessary.
2. Click the Data Catalog  icon in the application menu.
3. Click **Categories**.  
The system displays any existing categories.
4. Click the Delete  icon in the row for the category.
5. Click **Delete** to confirm this action.

## 6.4 See Also

- [Importing Schemas](#)
- [Editing Schemas](#)

# 7

## Introduction to Recipes and Staging Tables (2.9)

*Recipes* describe how to load data from external sources into InterSystems Data Fabric Studio™ or into external tables or files. When you define a recipe that loads data into InterSystems Data Fabric Studio™, the system generates staging tables and generates new versions of the tables in response to published changes in the data and in the recipe. This page provides an overview of a recipe and the generated staging tables.

### 7.1 Structure of a Recipe

Any recipe consists of some or all of the following steps, in order:

1. **Staging** activities. For each staging activity, you select a data source and then the schemas of interest from that data source.

When the recipe is run, the data for each schema is loaded into one or more generated **staging tables**. The system automatically uses the extraction strategy specified in the schema. In the recipe, you choose which fields to copy.

2. **Transformation** activities, which transform fields in the staging tables. You choose how to transform the value, and you specify whether to replace the existing value or save the transformed value into a new field that you specify (in the same table as the original value).
3. **Validation** activities, which examine the values in the staging table or tables. For each fields that you choose to validate, you specify the validation function.

The results of the validation checks are written to a validation report file. Validation errors can halt the recipe or can simply issue warnings, as you choose; in case of any validation error, the system generates a workflow task for the assigned role (specified when the recipe is **scheduled**).

4. **Reconciliation** activities, which compare values in different staging tables from the same run of the recipe. You specify how to match up records and you specify which fields to compare. Then for each comparison, you choose a comparison option.

The results are written to a reconciliation report file. Reconciliation errors can halt the recipe or can simply issue warnings, as you choose; in case of any reconciliation error, the system generates a workflow task for the assigned role (specified when the recipe is **scheduled**).

5. **Data promotion** activities. For each data promotion activity, you define a set of steps, each of which uses SQL to update a final table or file, based on the contents of the staging table or tables. If the target is a table, it can be in the native database or can be in an external database.

A recipe can also include custom steps at each stage in the processing.

## 7.2 Staging Tables

As a preliminary step, a recipe involves loading data into generated staging tables. The names of these tables are generated, and the table name format is as follows:

`Staging_recipeshort_stagingshort_vversion.schemaname`

Where *recipeshort* is the **Recipe Short Name**, *stagingshort* is the short name of the [staging](#) activity, *version* is the version of the recipe, and *schemaname* is the name of the schema as seen in the [Data Catalog](#). (Note that `Staging` is a default part of this name but is [configurable](#).)

Any staging table includes the following fields:

- Fields maintained and used internally by the system: `%IRISRowID` and `%BatchId`. The `%BatchId` field contains the ID of the batch in which the given record was updated; you use this to identify the records to use in the data promotion activity.
- A set of fields whose names match the field names from the data that is loaded into that table; these fields are automatically added as needed.
- Any additional fields defined within the [transformation](#) activities. These field names start with the prefix `%T_`.

As you create a recipe, you may find it helpful to [view](#) the structure of the staging table, particularly when you define the promotion activities.

## 7.3 See Also

- [Schema Evolution](#)
- [Defining Recipes](#)
- [Editing and Managing Recipes](#)
- [Scheduling and Running Tasks](#)
- [Viewing Run History](#) (includes information on viewing dropped records, validation reports, and reconciliation reports)


# 8

## Defining Recipes (2.9)

This page discusses how to create recipes. For related reference information, see [Activity Type Reference](#).

### 8.1 Creating a Recipe


To define a recipe:

1. Click the Recipes  icon in the application menu.
2. Click **New Recipe**.
3. For **Recipe Name**, enter a descriptive name to display in the Recipes Dashboard. This name cannot be changed later.
4. For **Recipe Short Name**, enter a short name to be used within the names of the [generated tables](#) used by this recipe. This name cannot be changed later.
5. For **Recipe Group**, optionally select the name of a [recipe group](#).  
You can move this recipe later to another group or remove it from the group.
6. Click **Submit**.

The system then displays a page where you can add activities to the recipe.

### 8.2 Adding a Staging Activity

A staging activity populates one or more staging tables. To add a staging activity to a recipe:

1. Display the recipe:
  - a. Click the Recipes  icon in the application menu.
  - b. If the recipe is in a group, expand that group.
  - c. Click the recipe name.  
The recipe is then displayed on the right side of the page.
2. Click **Add Staging Activity**.

3. For **Name**, enter a descriptive name to display within the recipe.
4. For **Short Name**, enter a short name to be used within the names of the [generated tables](#) used by this recipe. This name cannot be changed later.

If you are going to use only one data source in this recipe, you could use a generic short name like `Load`, but if you are going to use multiple data sources, it may be helpful for the short names to indicate the data source.

5. For **Data Source**, select a [data source](#).

This selection cannot be changed later.

6. Click **Submit**.

The system then displays a page where you can edit the details of how to use this data source. This page lists all the schemas that belong to this data source, and each schema represents a table, an specific API call, or a specific format of file. You can load data from any number of these items, within this staging activity.

7. For each schema that you want to include in this staging activity:
  - a. Click the schema name in the left. The right side of the page then displays information about the schema, including (if applicable) information about the last time data was loaded from the associated data source.
  - b. Specify how to use this data schema. First, optionally select the following check boxes:
    - **Stage on Target Table**—Enables you to copy data to the table name of your choice. The table is generated for you. If you select this, you are prompted for a full table name.
    - **Use SQLQuickload**—Enables the recipe to perform a bulk load for this step. If you select this, also specify the following options:
      - **Action On Dropped Records**—Specifies the action to take when the bulk load encounters more than the specified number of dropped records. The **Use SQLQuickload** option drops a record if that record cannot be inserted, which can occur if the record is invalid in some way (such as not having a required field or having a value in an incorrect format). For **Action On Dropped Records**, select one of the following, to specify how the system should behave if the maximum number of dropped records is reached:
        - **Abort**—The recipe is automatically aborted, without any workflow intervention.
        - **WorkflowStrict**—The recipe is stopped and a workflow task is generated. The workflow task provides the options **Abort** and **Retry**.
        - **Workflow**—The recipe is stopped and a workflow task is generated. The workflow task provides the options **Abort**, **Retry**, and **Ignore and Proceed**.
        - **Proceed**—The recipe continues.
      - **Max Dropped Records**—Specifies the maximum number of dropped records before the specified action is triggered.
  - c. Choose the fields to load. To do so, either click the check box at the top of the column of check boxes or click the check box for each field of interest.

8. When you are temporarily done with this activity, click **Save Draft**. Or if you are done defining the activity, click **Publish Activity**.

When you save the draft for the first time, the **Staging Table** field is updated to show the name of the [staging table](#). The table is not generated until you publish the activity. If you want to add additional activities that use the staging table, it is necessary to publish the staging activity.

9. Click the **Recipe** link in the upper left to return to the recipe.



## 8.3 Adding a Transformation Activity

A transformation activity transforms data in a staging table. To add a transformation activity to a recipe:

1. Display the recipe.
2. Make sure that you have published the staging activities for the staging tables you want to transform.
3. Click **Add Transformation Activity**.
4. For **Name**, enter a descriptive name to display within the recipe.
5. For **Target Source**, select a schema; the list includes all schemas used in *published* [staging activities](#) within the same recipe.

The system then displays the fields in the associated staging table, with options to specify transformations for any of the fields.

6. For each field where you want to apply a transformation, select the field name.

The system then displays options on the right side of the page. Make edits described below and then click **Apply**.

- **Function Type**—See [Transformation Functions](#).
- **Overwrite**—Select this if you want the new value to replace the old value. In this case, the system will not generate a new field in the staging table.

Otherwise, leave **Overwrite** cleared and specify a new field name (see below).

- **Transformation Name**—Specify a short name for this field transformation.
- **Result Field Name**—Specify a name for the new field that will store this transformed value. In this case, the system will automatically add a field to the staging table; the actual name of the new field is %T\_*newname* where *newname* is the name you specified.
- **Result Field Type**—Select the type for the new result field.
- Other options—See [Transformation Functions](#).

7. When you are done editing this activity, click **Save Draft**. Or click **Publish Activity** so that the recipe will use it.
8. Click the **Recipe** link in the upper left to return to the recipe.

## 8.4 Adding a Validation Activity

A validation activity validates data in a staging table and generates a validation report; the report will flag any rows that failed the validation rules that you specify. For each validation rule, you specify whether to halt the recipe or to simply issue a warning. In the case of any validation error, the system generates a workflow task for the assigned role (specified when the recipe is [scheduled](#)).

To add a validation activity to a recipe:

1. Display the recipe.
2. Click **Add Validation Activity**.
3. For **Name**, enter a descriptive name to display within the recipe.

4. For **Target Source**, select the schema. The list includes all schemas used in *published* [staging activities](#) within the same recipe.

The system then displays the fields in the staging table, with options to specify validation rules for any of the fields. Note that any field can have at most one validation rule, within a given validation activity.

5. For each field where you want to apply a validation rule, select the field name.

The system then displays options on the right side of the page. Make edits described below and then click **Apply**.

- **Function Type**—See [Validation Functions](#).
- **Validation Name**—Specify a short name for this field transformation.
- **Result Field Name**—Specify a name for the report field that will store either 1 (if the field value passes the validation rule) or 0 (if it does not). It is useful for the name to indicate the field being examined, for example `CheckItemCount`.
- **Validation Failure**—Specify how the system should respond when it encounters a field value that fails this specific rule. A **Fatal** failure halts the recipe. A **Warning** failure logs a warning. Both kinds of failures are visible in the [Workflow Inbox](#).
- Other options—See [Validation Functions](#).

6. When you are done editing this activity, click **Save Draft**. Or click **Publish Activity** so that the recipe will use it.

7. Click the **Recipe** link in the upper left to return to the recipe.

## 8.5 Adding a Reconciliation Activity

A reconciliation activity compares data in a staging table to comparison data, which can be either data that you have loaded (and if needed transformed and validated) within the same recipe or data from another data source. This activity generates a reconciliation report.

To add a reconciliation activity to a recipe:

1. Display the recipe.
2. Click **Add Reconciliation Activity**.
3. For **Reconciliation Activity Name**, enter a descriptive name to display within the recipe.
4. In the **Primary Source** section, specify the table that you ultimately want to promote and use downstream. To do so:
  - a. For **Data Source**, select the location of the table that you want to use. Typically this is **Staging Tables**, because the most common use case is to reconcile data you have loaded within a recipe.
  - b. For **Tables**, select the table.
5. In the **Secondary Source** section, specify the comparison table. To do so:
  - a. For **Data Source**, select the location of the table that you want to use.
  - b. For **Tables**, select the table.
6. Click **Create Activity**.
7. On the **Key Map Definition** tab, optionally click the **Report Missing Records from Primary Source (Full Outer Join)** check box. If you select this check box, the comparison will report whenever there are records in the primary source that are not found in the secondary source.

Below this check box, the page initially displays a list of the fields in the primary source, with no information about how these are to be matched to fields in the secondary source.

8. The next step is to specify how to match a row in the primary source to a row in the secondary source. Typically, each source has a primary key (consisting of one or more fields), and the primary keys can be matched to each other; that is a record that has a specific primary key in the one source will be matched to the record that has the same primary key in the second source.

If the sources both have a primary key, the easiest approach is to click **Auto Match Primary Keys**. When you do so, the **Key Map** area shows the primary key fields from the primary source on the left, with the primary key fields from the secondary (comparison) source on the right.

9. Now add fields to the comparison, as follows:

- a. Click a field name under **Primary Source**—this adds the field to the left column in **Key Map** area.
- b. Look for the field under **Secondary Source** that should be used for comparison for this field. Click that field name. The system adds this field to the right column in **Key Map** area.

Or click **Try to Match Selected Column**. In this case, the system looks for an appropriately named field in the secondary source.

To remove an item in the **Key Map** area, click its X button.




10. When you are done adding fields, click **Create**. The system then creates the key map for this reconciliation activity. If needed, you can return later and make edits.
11. Click **Reconciliation**.

Now the page displays a table listing the fields from the two sources. Here you specify rules used to compare the values.

12. Briefly review the table for familiarity. The **Primary Field** column lists the fields from the primary source, and the **Secondary Field** column list the fields from the secondary source. This information is the same as in the key map. In addition, note that **Data Type** indicates the data types of the fields from the primary source (which is helpful in choosing a comparison function).
13. If many or most of the fields have the same name in the two tables, click **Quick Match All** to quickly create a set of reconciliation rules that you can then review and edit. With this option, the system automatically creates a rule for each pair of fields that have the same name (including case). Each automatically generated rule includes a unique rule name, a comparison function, a generated field name (to hold the comparison result), and (for numeric comparisons), an initial setting for **Tolerance**.

Some fields may not yet have rules (which is OK).

14. Make edits (as needed) for each rule:

- a. In **Function Type**, choose a comparison function.
- b. For **Rule Name**, click the Edit  icon and then type a unique rule name.
- c. For **Result Field**, click the Edit  icon and then type a unique field name. The reconciliation activity will write a 1 or a 0 to this field, to indicate whether reconciliation was successful.
- d. For **Tolerance** (required for numeric comparison), click the Edit  icon and then type the numeric tolerance.  
If the tolerance ends with a percent sign (%), then the system treats it as a percentage.
- e. If failing this reconciliation check should be a fatal error (halting the recipe), select the **Fatal** check box.

15. When you are done editing this activity, click **Save Draft**. Or click **Publish Activity** so that the recipe will use it.

16. Click the **Recipe** link in the upper left to return to the recipe.

## 8.6 Viewing the Staging Table Details

Before you can add a promotion activity, it is helpful to review the structure of the [staging table or tables](#) used by the recipe. In particular, make a note of the field names that you need to use. To see the definition of a staging table:

1. Display the recipe.
2. At the bottom of the page, see the list in **Indices and Staging Table Details**.  
This section shows one row for each data source used in the recipe.
3. Click the row for a given data source. The page then lists the generated staging tables associated with this data source.
4. Click a table name in this list. The right side of the page then provides information about the indices on the table, along with details about the fields in this table.
5. Optionally add indices as needed. To add an index, click **New Index** and then specify an index name and select one or more properties to be indexed.

## 8.7 Adding a Promotion Activity

A promotion activity copies data from staging tables to their final targets, which can be any of the following:

- A table in Data Fabric Studio
- A table in an external database
- A file in an S3 bucket

The activity consists of one or more promotion items, each of which uses custom SQL to delete, insert, or update records.

Because the staging table name is generated, it is necessary to refer to the staging table name in an indirect way. Similarly, it is necessary to refer indirectly to the current run of the recipe. Accordingly, the product provides options that insert special tokens that you can use within your SQL; these tokens are automatically replaced when the recipe is run.

To add a promotion activity to a recipe:

1. Display the recipe.
2. Click **Add Promotion Activity**.
3. For **Name**, enter a descriptive name to display within the recipe.
4. For **Run Order**, select or type an integer, which controls the order in which the promotion activities are run, with respect to other promotion activities in the same recipe.
5. For **Promotion Type**, select one of the following:
  - **Internal** — Select this to load data into a table within this database (that is, the database native to Data Fabric Studio).
  - **External** — Select this to load data into a table in another database or a file in an S3 bucket.

If you select **External**, also specify **Target Connection**. This must be either the name of an external database or an S3 bucket, in either case previously configured as a [data source](#).

6. Click **Submit**.
7. Click **New Item**.
8. Specify the item as described in the [next section](#).
9. Click **Save Activity Item**.
10. If necessary, add other items to this promotion activity.
11. When you are done editing this activity, click **Save Draft**. Or click **Publish Activity** so that the recipe will use it.

For each promotion activity item, the system compares the number of source fields to the number of target fields; if these numbers do not match, the system displays an error and does not let you publish the activity. Similarly, the system compares the data types of the data being promoted to the data types of the fields in the target. If the data types do not match, the system displays an error and does not let you publish the activity.

12. Click the **Recipe** link in the upper left to return to the recipe.

## 8.8 Adding a Promotion Activity Item

A [promotion activity](#) consists of one or more activity items. For each activity item, specify the following values:

- **Run Order**—Specify the order in which this item should be executed, relative to other items in the same promotion activity. Select an integer.
- **Description**—Enter an optional description of this item, as an aid to yourself and others reviewing this recipe later.
- **Operation Type**—Choose one of the following:
  - **Insert**
  - **Insert or Update on Primary Key**
- **SQL Expression**—Enter an SQL UPDATE, INSERT, or DELETE statement.

The right side of the page provides samples to assist you in writing the query. When you are getting started, it is a good idea to review these and find an example that matches your scenario, to use as a starting place.

Below **SQL Expression**, the page provides options that you can use to insert helpful tokens into the query. To insert a reference to a staging table:

1. Place the cursor within **SQL Expression** at the position where you need the staging table reference.
2. For **Placeholders**, select **Staging Activity**.
3. For **Staging Activities**, select the short name of the staging activity to which the desired staging table belongs. (For example, `mystage`.)
4. For **Data Source Items**, select the data schema to which the desired staging table belongs. (For example, `samplecsv`.)
5. Click **Add to SQL**.

This inserts the following string into your query:

```
{mystage}.samplecsv
```

This is a generic reference to the staging table for the `samplecsv`, generated and populated by the `mystage` staging activity within this recipe. When the recipe is run, this token is replaced with the actual name of this staging table.

Similarly, when selecting records from the staging table (to insert or update the final table or file), you typically select only the records from the current run of the recipe. To modify your query to do this:

1. If there is no **WHERE** clause yet, move the cursor to the end of **SQL Expression** and type **WHERE** followed by a space.
2. For **Placeholders**, select **BatchID**.
3. Click **Add to SQL**.

This inserts the following string into your query:

```
%BatchId={%BatchId}
```

Note that `%BatchId` is the name of the field that stores the recipe run ID, and the token `{%BatchId}` is converted automatically into the ID of the current recipe run.

You can also type these syntaxes directly into the query.

## 8.8.1 Specifying the Target File Settings

If this promotion activity creates a target file, click the **Target File Settings** tab and specify the following values:

- **Target subfolder**—Specifies the subfolder into which the target file should be written. This option is optional. If you omit this value, the file is written to the main directory.
- **Target Filename Pattern**—Specifies the name of the file to create. The filename can include the `%RUNDATE` token, which is replaced with the date and time when the file is created.
- **Delimiter**—Specifies the field delimiter. This must be a single ASCII character. To specify a control character, wrap the name of the control character in angle brackets.
- **Line Terminator**—Specifies the line terminator to use when creating this file. Use a value appropriate for the target operating system.

## 8.9 Adding a Custom Activity

A recipe can include custom activities. For example, it may be necessary to have a custom activity that retrieves the data from a data source (for example, to place files within the file system). In such cases, two items are needed:

- Custom code in the form of a business host added within the interoperability engine that system uses internally. The business host must be a business operation or a business process that accepts as input a message of type `SDS.DataLoader.CustomActivityRequest`. Details are currently beyond the scope of the documentation.
- Configuration of the recipe to use that code at the applicable phase. To configure a recipe to use custom code:
  1. Display the recipe.
  2. Click **Add Custom Activity**. This displays a dialog box where you specify the details.
  3. For **Name**, enter the name of this activity.
  4. For **Target Business Host**, enter the configuration name of the business host.
  5. For **When to Run**, select **Before Staging**, **Before Transformation**, **Before Validation**, **Before Reconciliation**, **Before Promotion**, or **After Promotion**.
  6. For **Enabled**, select this check box if the activity should be enabled.
  7. Click **Submit**.

## 8.10 Step Mode

You can run a recipe in step mode, a mode that lets you use a subset of data—perhaps a very small subset—so that you can test the recipe quickly from end to end. In this mode, you specify the number of rows of data to load into the staging tables (and then to use in the subsequent processing).

To configure a recipe for step mode:

1. Display the recipe.
2. Click the **Step Mode** toggle.
3. In the box next to **Step Mode**, select or type the number of records to use.

A recipe remains in step mode until you change the **Step Mode** toggle back to off.

## 8.11 See Also

- [Introduction to Recipes and Staging Tables](#)
- [Activity Type Reference](#)
- [Editing and Managing Recipes](#)
- [Managing Recipe Groups](#)
- [Scheduling and Running Tasks](#)
- [Viewing Run History](#) (includes information on viewing dropped records, validation reports, and reconciliation reports)





# Activity Type Reference (2.9)

This section provides reference information for each activity type in [recipes](#).

## Staging Activity (2.9)

---

Loads data from one or more data schemas.

### Introduction

A [staging activity](#) loads data from one or more [data schemas](#), loading data for each schema to its own generated staging table. For each schema, you can control which fields are copied, as well as how the load operation is performed. Within a staging activity, all schemas must belong to a single data source.

### Options

For each data schema used in a staging activity, the following options apply:

- **Primary Key Fields**—Indicates the primary key fields of the schema. This option is controlled by the schema definition and cannot be changed within a recipe.
- **Extraction Strategy**—Indicates how data is extracted from this schema. This option is controlled by the schema definition and cannot be changed within a recipe.
- **Staging Table**—Specifies the table into which this activity loads data. This table name is generated and cannot be changed.
- **Stage on Target Table**—Enables you to copy data to the table name of your choice. The table is generated for you. If you select this, you are prompted for a full table name.
- **Use SQLQuickload**—Enables the system to perform a bulk load for this step. If you select this, also specify the following options:
  - **Action On Dropped Records**—Specifies the action to take when the bulk load encounters more than the specified number of dropped records. The **Use SQLQuickload** option drops a record if that record cannot be inserted, which can occur if the record is invalid in some way (such as not having a required field or having a value in an incorrect format). For **Action On Dropped Records**, select one of the following, to specify how the system should behave if the maximum number of dropped records is reached:
    - **Abort**—The recipe is automatically aborted, without any workflow intervention.
    - **WorkflowStrict**—The recipe is stopped and a workflow task is generated. The workflow task provides the options **Abort** and **Retry**.
    - **Workflow**—The recipe is stopped and a workflow task is generated. The workflow task provides the options **Abort**, **Retry**, and **Ignore and Proceed**.
    - **Proceed**—The recipe continues.
  - **Max Dropped Records**—Specifies the maximum number of dropped records before the specified action is triggered.

### See Also

- [Introduction to Recipes and Staging Tables](#)
- [Adding a Staging Activity](#)

# Transformation Activity (2.9)

Transforms values in a staging table.

## Introduction

A [transformation activity](#) transforms fields in a single staging table. For each field that is transformed, you can either overwrite the existing field or set the value of a new field.

## Options

For each field to be transformed, the following options apply:

- **Function Type**—Specifies the function to use to transform the value. See [Transformation Functions](#).
- **Overwrite**—Specifies whether to replace the old value. In this case, the system will not generate a new field in the staging table.

If **Overwrite** is cleared, the system writes the new value into the field named by **Result Field Name**.

- **Transformation Name**—Specifies a short name for this field transformation.
- **Result Field Name**—Specifies field that will store this transformed value (if **Overwrite** is not selected). The system automatically adds a field to the staging table; the actual name of the new field is %T\_*newname* where *newname* is the name you specified.
- **Result Field Type**—Specifies the type for the new result field.

There are additional options depending on **Function Type**; see the next section.

## Transformation Functions

The available transformation functions are as follows:

### Code Table Conversion

*Available for string fields.* This function transforms the data by using a code table, which provides a set of key/value pairs. If the value being converted matches a key in the given code table, that value is converted to the value associated with that key. Also specify:

- **Code Table Name**—Specify the name of a table that contains key/value data.
- **Code Table Key Column**—Specify the name of the field in that table that contains the keys.
- **Code Table Value Column**—Specify the name of the field in that table that contains the associated values.

### Custom SQL Expression

*Available for string fields.* This function transforms the data by using an SQL expression. Also specify **SQL Expression** as an SQL expression that returns a single value, such as an expression that combines field names and SQL operators. You must delimit any field name that is an SQL reserved word.

### Lowercase String

*Available for string fields.* This function converts the string to lowercase.

### Replace Substring

*Available for string fields.* This function performs a substring replacement. Also specify:

- **Old Substring**—Type the text to be replaced.

- **New Substring**— Type the replacement text.

#### **Trim Whitespace**

*Available for string fields.* This function trims whitespace (leading, trailing, or both). Select **Trim Leading Whitespace**, **Trim Trailing Whitespace**, or both.

#### **Trim Characters**

*Available for string fields.* This function trims a specified set of characters (leading, trailing, or both). Specify one or more characters for **Characters To Trim**. Then select **Trim Leading Characters**, **Trim Trailing Characters**, or both.

#### **Uppercase String**

*Available for string fields.* This function converts the string to uppercase.

#### **Round Number**

*Available for numeric fields.* This function rounds the number as specified. Also specify **Scale**, the number of decimal places.

## **See Also**

- [Introduction to Recipes and Staging Tables](#)
- [Adding a Transformation Activity](#)

# Validation Activity (2.9)

Validates the fields in a staging table and generates a report on any validation failures.

## Introduction

A [validation activity](#) examines a staging table and validates the fields in it. For each field, the options depend on the data type and the function used for validation. For each field, you can also specify how to handle a validation failure.

The results of the validation checks are written to a validation report file.

## Options

For each field to be validated, the following options apply:

- **Function Type**—Specifies how to validate the field; see [Validation Functions](#).
- **Validation Name**—Specifies a short name for this field transformation.
- **Result Field Name**—Specifies a name for the report field that will store either 1 (if the field value passes the validation rule) or 0 (if it does not). It is useful for the name to indicate the field being examined, for example `CheckItemCount`.
- **Validation Failure**—Specifies how the system should respond when it encounters a field value that fails this specific rule. A **Fatal** failure halts the recipe. A **Warning** failure logs a warning. Both kinds of failures are visible in the [Workflow Inbox](#).

There are additional options depending on **Function Type**; see the next section.

## Validation Functions

The available validation functions are as follows:

### Code Table

*Available for string and numeric fields.* For this function, the field value fails validation if it does not match a value in a code table. Also specify:

- **Code Table Name**—Specify the name of the code table to use.
- **Code Table Key Column**—Specify the name of the key column in this table.

### Custom SQL Expression

*Available for all types of fields.* For this function, the field value fails validation if an SQL predicate expression returns false. (You can, for example, compare the value in this field to the value in another field in the same record.) Also specify:

- **SQL Operator**—Select a comparison operator.
- **SQL Expression**—Enter a scalar SQL expression such as a field name or a constant.

### Glob Pattern Matching

*Available for string fields.* For this function, the field value fails validation if it does not match the given pattern. Also specify:

- **Glob Pattern**—Specify a glob pattern.
- **Escape Character**—Specify an escape character.

This function uses the InterSystems SQL [%MATCHES predicate](#).

#### IRIS Pattern Matching

*Available for string fields.* For this function, the field value fails validation if it does not match the given pattern. Also specify **Pattern** This function uses the InterSystems SQL [%PATTERN predicate](#).

#### Matches Regular Expression

*Available for string fields.* For this function, the field value fails validation if it does not match the given regular expression. Also specify **Regular Expression** This function uses the ObjectScript [\\$MATCH function](#).

#### Does Not Match Regular Expression

*Available for string fields.* For this function, the field value fails validation if it *does* match the given regular expression. Also specify **Regular Expression** This function uses the ObjectScript [\\$MATCH function](#).

#### Not Null

*Available for all types of fields.* For this function, the field value fails validation if it is null.

#### Numeric Change Within Range

*Available for numeric, date, and time fields.* For this function, the field value fails validation if the value obtained in this recipe run is too different from the value in the previous run of the recipe. Also specify **Tolerance**—the amount by which the value is permitted to be different from the previous value. This can be either an absolute value or a percentage. (To specify a percentage, include a percent sign at the end.)

If the field is a date field, the unit is a day. If the field is a datetime field, the unit is a second.

#### Fixed Numeric Range

*Available for numeric fields.* For this function, the field value fails validation if it falls outside of a fixed range of numeric values. For the **Fixed Numeric Range** function type, also specify:

- **Minimum**— Minimum permitted numeric value.
- **Maximum**— Maximum permitted numeric value.
- **Tolerance**—Specify the tolerance, or the amount by which the value is permitted to be outside of the specified range. This can be an absolute value or a percentage. (To specify a percentage, include a percent sign at the end.)

#### String Change Within Code Table

*Available for string fields.* This function is intended for the scenario where there is a code table that represents an ordered sequence such as a set of ratings, and the goal is to check that the field value has not changed too much within that sequence, since the previous successful load of data. For example, in this code table, the rating AAA may be represented as 10, the rating AA+ may be represented as 9, and so on. To use this function type, also specify:

- **Tolerance**—Specify the tolerance, or the amount by which the value is permitted to be outside of the specified range. This can be an absolute value or a percentage. (To specify a percentage, include a percent sign at the end.)
- **Code Table Name**—Specify the name of the code table to use.
- **Code Table Key Column**—Specify the name of the key column in this table. The key values are treated as strings.

- **Code Table Value Column**—Specify the name of the value column in this table. The values in this column must be numeric.

This function examines the previous and current values of the field, both of which are strings such as AAA and AA+, converts them to the corresponding numbers, and compares the numbers, taking the tolerance into account. If the numbers match or are within the tolerance amount of each other, the new field value passes validation.

#### **Value Changed Since Previous**

*Available for all types of fields.* For this function, the field value fails validation if the value is different from the value obtained in the previous successful run of the recipe. Use this rule if you expect the value to always be the same.

## **See Also**

- [Introduction to Recipes and Staging Tables](#)
- [Adding a Validation Activity](#)

## Reconciliation Activity (2.9)

---

Compares the most recent data loaded into two tables and generates a report on any differences.

### Introduction

A reconciliation activity compares the most recent data loaded into two tables. The key map describes how to match a record in one table (the primary table) with a record in the other table (the secondary table). Creating the key map consists of two steps:

1. Identifying the primary key of each table. Then for a given record in the primary table, the system knows which record of the secondary table to use as comparison.
2. Matching up the other fields as appropriate. Then for a given field in a given record in the primary table, the system knows which field of the other record to use as comparison.

For each field-to-field comparison, a reconciliation activity specifies a comparison function, a tolerance (if applicable), and how to handle a reconciliation failure.

The results of the reconciliation checks are written to a reconciliation report file.

### Options

For each field to be reconciled, the following options apply:

- **Function Type**—Specifies the comparison function to use.
- **Rule Name**—Specifies a unique rule name.
- **Result Field**—Specifies a unique field name. The reconciliation activity will write a 1 or a 0 to this field, to indicate whether reconciliation was successful.
- **Tolerance**—Specifies the numeric tolerance. This option is required for numeric comparisons.  
If the tolerance ends with a percent sign (%), then the system treats it as a percentage.
- **Fatal**—If selected, that means that failing this reconciliation check should be a fatal error (halting the recipe).

### See Also

- [Introduction to Recipes and Staging Tables](#)
- [Adding a Reconciliation Activity](#)



# Promotion Activity (2.9)

Promotes data to the final tables or files.

## Introduction

A [promotion activity](#) copies data from staging tables to their final tables or files, where they can be used by downstream consumers. A promotion activity consists of one or more promotion items, which are to be performed in a specific order. Each promotion item copies data to a final table or file. The data copied can use data from multiple staging tables.

## Options

For each promotion item, the following options apply:

- **Promotion Type**—Specifies whether the data is to be promoted to a table within this database (that is, the database native to Data Fabric Studio) or to a table in another database. The choices are as follows:
  - **Internal**—This option loads data into a table within this database.
  - **External**—This option loads data into a table in another database or to an external file.

After you create an item in a promotion activity, you cannot change this setting, and this option is no longer displayed.

- **Target Connection**—*Applies if target is external.* Specifies the database or external file system to promote the data into.

After you create an item in a promotion activity, you cannot change this setting.

- **Run Order**—Specifies the order in which this item should be executed, relative to other items in the same promotion activity. Select an integer.
- **Description**—Specifies an optional description of this item, as an aid to yourself and others reviewing this recipe later.
- **Operation Type** *Applies if target is external.*—Choose one of the following:
  - **Insert**
  - **Insert or Update on Primary Key**
- **SQL Expression**—Specifies an SQL UPDATE, INSERT, or DELETE statement.
- **Target subfolder** *Applies if target is a file.*—Specifies the subfolder into which the target file should be written. This option is optional. If you omit this value, the file is written to the main directory.
- **Target Filename Pattern** *Applies if target is a file.*—Specifies the name of the file to create. The filename can include the %RUNDATE token, which is replaced with the date and time when the file is created.
- **Delimiter** *Applies if target is a file.*—Specifies the field delimiter. This must be a single ASCII character. To specify a control character, wrap the name of the control character in angle brackets.
- **Line Terminator** *Applies if target is a file.*—Specifies the line terminator to use when creating this file. Use a value appropriate for the target operating system.

## See Also

- [Introduction to Recipes and Staging Tables](#)
- [Adding a Promotion Activity](#)

## Custom Activity (2.9)

---

Performs custom processing at any phase during a recipe.

### Introduction

A recipe can include custom activities. For example, it may be necessary to have a custom activity that retrieves the data from a data source (for example, to place files within the file system). In such cases, two items are needed:

- Custom code in the form of a business host added within the interoperability engine that system uses internally. The business host must be a business operation or a business process that accepts as input a message of type `SDS.DataLoader.CustomActivityRequest`. Details are currently beyond the scope of the documentation.
- Configuration of the recipe to include a custom activity that calls this code.

### Options

For each custom activity, the following options apply:

- **Target Business Host**—Specifies the configuration name of the business host that performs the custom processing.
- **When to Run**—Specifies when this processing should occur. The choices are as follows:
  - **Before Staging**
  - **Before Transformation**
  - **Before Validation**
  - **Before Reconciliation**
  - **Before Promotion**
  - **After Promotion**
- **Enabled**—Specifies whether the recipe performs this activity.

### See Also

- [Introduction to Recipes and Staging Tables](#)
- [Adding a Custom Activity](#)

# 9

## Editing and Managing Recipes (2.9)

In InterSystems Data Fabric Studio™, a recipe describes how to load data into the system. This page describes how to edit and manage recipes; another page provides information on [defining](#) them.

For convenience, recipes can be in [groups](#).


### 9.1 Recipe Changes

One of the key objectives is to permit schemas to change, while preserving your data. [Schema evolution](#) is described in detail elsewhere, but note the following:

1. When you [edit](#) or [reimport](#) a schema, you are implicitly creating a draft that is not yet used by anything. The existing schema is unchanged; existing data is also unchanged.
2. When you [republish](#) a schema, *if* the change affects the structure needed to contain the data, the system automatically increments the version number of the schema and automatically generates a new draft of any recipe staging activity that is affected by this change. The new staging activity draft is not automatically published, and the previous staging activity is unchanged.
3. When you republish any draft staging activities, the system generates a new staging table, considering the structural change that has occurred. This leaves the old table (and its data) unchanged.

Independent from any schema changes, a recipe can also require fields to be added to the staging table. The system handles this automatically.

### 9.2 Viewing the Recipes

To see all the available recipes, click the Recipes  icon in the application menu. The page displays a list of the recipes; some or all of them might be in [groups](#). In this display, the groups are listed alphabetically, followed by any recipes that are in no groups. Within a group, the recipes are listed alphabetically. You can [specify the group](#), if any, to which a recipe belongs.

Because you might have a large number of recipes, this page provides filter options that can help you find a recipe more quickly.


To display only recipes whose names match a given string, type that string into the Search box. Note that this hides recipes whose names do not match the string, but does not hide any groups.

To filter the recipes so that you see only the recipes with alerts or notifications, click the Filter icon and then click **Only Show Items With Alerts/Notifications**. Then click **Apply**.

This page does not show deleted groups by default. To display deleted groups, click the Filter icon and then click **Show Deleted Groups**. Then click **Apply**.

## 9.3 Editing a Recipe

To edit a recipe (other than [renaming](#) it):

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the recipe. The system then displays the current definition, including any draft activities.

If there is a draft activity, you can choose to display the published version or the draft version (which is shown as a separate item as in the following example).

Staging Activities				
	Name	Short Name	Data Source	Delta Version
	LoadCSV	LoadCSV	filedir	2
DRAFT	LoadCSV	LoadCSV	filedir	3

Now you can make any of the following edits:

- To add a new activity, click **Add Staging Activity**, **Add Transformation Activity**, **Add Validation Activity**, **Add Reconciliation Activity**, or **Add Data Promotion Activity**, and then continue as described in [Defining Recipes](#).

The new activity will not take effect until you publish it via the **Publish Activity** button.


- To edit an existing activity, click it. Notes:
  - If you click a draft that the system generated because of [schema evolution](#), the system displays a message that explains why it generated this new draft.
  - If you are viewing a published activity, you must click **Edit Activity** to make it editable.
  - If you are viewing a published activity *and* there is a current draft for that activity, the **Edit Activity** button displays the draft in edit mode.

Make changes as needed and save them via the **Save Draft** button.

The changes will not take effect until you publish it via the **Publish Activity** button.


## 9.4 Publishing an Activity

To publish (or republish) an activity within a recipe:

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the recipe. The system then displays the current definition, including any draft activities.  
If there is a draft activity, you can choose to display the published version or the draft version.
4. Click the activity that you want to republish.
5. Click **Publish Activity**.

## 9.5 Disabling an Activity

To disable an activity within a recipe (so that this activity is skipped if the recipe is used):

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the recipe. The system then displays the current definition, including any draft activities.  
If there is a draft activity, you can choose to display the published version or the draft version.
4. Click the activity that you want to disable.
5. Click the slider in the **Actions** column for that activity. Then click **Disable Current Activity**. Here's an example (before):


Disabled	Actions
No	<input type="checkbox"/>

After:

Disabled	Actions
Yes	<input type="checkbox"/>

## 9.6 Enabling an Activity

To enable an activity that has previously been disabled:


1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the recipe. The system then displays the current definition, including any draft activities.

If there is a draft activity, you can choose to display the published version or the draft version.

4. Click the activity that you want to enable.
5. Click the slider in the **Actions** column for that activity.

## 9.7 Deleting an Activity

To delete an activity in a recipe.



1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the recipe. The system then displays the current definition, including any draft activities.
4. Click the activity that you want to delete.
5. Click **Delete**. (Or if you are deleting a draft activity, click **Delete Draft**.)

Note that system-generated drafts cannot be deleted; system-generated drafts are created when, for example, an associated schema has changed.

6. Click **Delete** again.



## 9.8 Renaming a Recipe

To rename a recipe:

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the Menu  icon next to the recipe name.
4. Click **Rename**.
5. Enter new values as needed for **Recipe Name** and **Short Name**.
6. Click **Rename** to save the changes.

## 9.9 Moving a Recipe Into or Out of a Group

You can move a recipe into a group or out of all groups via drag and drop. The system also provides a menu option, as follows:



1. Click the Recipes  icon in the application menu.
2. If the recipe is currently in a group, expand that group.
3. Click the Menu  icon next to the recipe name.

4. Click **Move to other group**.
5. Select a group name (or select **Ungrouped** if this recipe should not be in any group).

Note that you cannot move a recipe into a group that has been deleted.



## 9.10 Copying a Recipe

To copy a recipe:

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the Menu  icon next to the recipe name.
4. Click **Copy**.
5. Enter new values as needed for **Recipe Name** and **Short Name**.
6. Optionally select **Continue Incremental Loads**. This option is useful if the new recipe is intended to carry on the same work as the one it was copied from. If this option is selected and if the original recipe loaded data incrementally and its last load was successful, that data will be checked and the new recipe will pick up where the original left off.
7. Click **Copy Recipe** to save the changes.

## 9.11 Disabling a Recipe


To disable a recipe, so that it cannot be run:


1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the Menu  icon next to the recipe name.
4. Click **Disable**.
5. Click **Disable** to confirm.

When you disable a recipe, the system disables any scheduled tasks that run the recipe.

## 9.12 Enabling a Recipe



To enable a recipe that has previously been disabled:

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.

3. Click the Menu  icon next to the recipe name.
4. Click **Enable**.

## 9.13 Purging Old Recipe Data

To purge old data for a recipe:

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the Menu  icon next to the recipe name.
4. Click **Reset**.
5. Click one of the following options:
  - **Full Reset** deletes all staging tables (both old and current) and deletes the recipe run history.
  - **Reset All But Current** just deletes the old staging tables, leaving the current staging tables as is. The old staging tables correspond to previous versions of the recipe.

Neither option affects the *definitions* of the current staging tables.



When you click an option, the right side of the dialog box provides additional detail on what will be deleted.

6. Click **Reset Recipe** to confirm.

**CAUTION:** This step is irreversible.

## 9.14 Deleting a Recipe

To delete a recipe:


1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the Menu  icon next to the recipe name.
4. Click **Delete**.
5. Click **Delete** to confirm.

When you delete a recipe, the system disables any scheduled tasks that run the recipe.

## 9.15 Recovering a Recipe

To recover a recipe that was previously deleted:





1. Click the Menu  icon next to the recipe name.
2. Click **Recover**.

The recipe is immediately restored and now can be used.

## 9.16 Permanently Deleting a Recipe

When you delete a recipe, it is possible to recover it. You can also permanently delete a recipe (so that it cannot be recovered). To do so:

1. [Delete the recipe](#) as previously described or [delete the group](#) to which it belongs.
2. If you deleted the group:
  - a. Click the Filter  icon.
  - b. Then click **Show Deleted Groups** and then click **Apply**.
3. Click the Menu  icon next to the recipe name.
4. Click **Permanently Delete**.
5. Click **Delete** to confirm.

When you permanently delete a recipe, the system also deletes the following items:

- Any scheduled tasks that run the recipe.
- All run history associated with the recipe.
- The staging tables that were generated for the recipe.

## 9.17 See Also

- [Defining Recipes](#)
- [Managing Recipe Groups](#)
- [Scheduling and Running Tasks](#)
- [Handling Task Errors](#)
- [Viewing Run History](#)




# 10

## Managing Recipe Groups (2.9)

Ultimately there may be a large number of [recipes](#), so the product provides an option to group them. When you display the recipes, the groups are listed alphabetically, followed by any recipes that are in no groups. Within a group, the recipes are listed alphabetically.



### 10.1 Creating a Recipe Group

To create a recipe group:

1. Click the Recipes  icon in the application menu.
2. Click **New group**.
3. Type a unique group name and press **Return** (or click the check mark).

### 10.2 Renaming a Recipe Group

To rename a recipe group:



1. Click the Recipes  icon in the application menu.
2. Click the Menu  icon next to the recipe group name.
3. Click **Rename**.
4. Type the new name and click the check mark.

The new name must be unique.

## 10.3 Deleting a Recipe Group




**Tip:** Before deleting a recipe group, identify any recipes in the group that you want to keep, and move those out of this group. When you delete a recipe group, that action deletes the recipes in it. They can be recovered after that step, but the easiest procedure is to move them first.

To delete a recipe group:

1. Click the Recipes  icon in the application menu.
2. Click the Menu  icon next to the recipe group name.
3. Click **Delete**.
4. Click **Delete Group** to confirm this action.

## 10.4 Recovering a Deleted Recipe Group

To recover a recipe group that was previously deleted:

1. Click the Recipes  icon in the application menu.
2. Click the Filter  icon.
3. Click **Show Deleted Groups** and then click **Apply**.  
This action displays the deleted groups.
4. Click the Menu  icon next to the recipe group name.
5. Click **Recover**.
6. Click one of the following options, depending on whether you also want to recover the recipes in this group:
  - **Recover the group only**
  - **Recover the group and all of the group's recipes**

Then click **Submit**.

Note that you can also [recover individual deleted recipes](#).

## 10.5 See Also

- [Defining Recipes](#)
- [Editing and Managing Recipes](#)

# 11

## Defining Snapshots (2.9)

*Snapshots* are static copies of data in InterSystems Data Fabric Studio™, for future review. This page describes how to define them; [another page](#) describes how to edit and manage them.

Once you have defined snapshots, you can [run them](#) (loading data into the system), either on a schedule or manually.

### 11.1 Uses of Snapshots

With snapshots, you can easily save data for later inspection by regulators; the snapshot can pull data from multiple tables as needed, and the system applies a tag to the records.

When you have run a snapshot multiple times, applying a different tag each time, you can examine how that data changes over time. In particular, you can build a cube on the snapshot data, using the tag values as a dimension.

### 11.2 Snapshot Tables

For each snapshot, the system will generate a table. The table name is based on the snapshot definition, and the columns are based on the query used in the snapshot definition. The table name format is as follows:


`halp_snapshot_snapshot.tablenameinteger`

Where *snapshot* is the **Short Name** of the snapshot and *tablename* is the short table name, both of which you provide when defining the snapshot. At the end of the name, *integer* indicates the version number. (Note that `halp_snapshot` is a default part of this name but is [configurable](#).)

If you change a snapshot definition so that it defines a different set of columns, the system automatically increments the version number and generates a new table that has the required columns. For example, when you define a snapshot, the generated table name may be `halp_snapshot_Sample.SampleTable1`, but if you redefine the query, the new generated table name becomes `halp_snapshot_Sample.SampleTable2`, and so on.

### 11.3 Defining a Snapshot

To define a snapshot:

1. Click the Snapshot  icon in the application menu.
2. Click **New Snapshot**.  
The system displays a dialog box where you specify initial information.
3. Specify the following information:
  - **Name**—*Required*. Specify a unique name for this snapshot.
  - **Short Name**—*Required*. Specify a unique short name for this snapshot, to be used within the fully qualified name of the generated table.
  - **Table Name**—*Required*. Specify a unique short table name (also to be used within the fully qualified name of the generated table).
  - **Tag**—*Required*. Specify a short string that becomes a tag applied to the snapshot when the snapshot is executed. As an example, a tag could indicate the state of the data, such as Preliminary or Final.
  - **Description**—Type a description of this snapshot and its purpose.

Except for **Short Name** and **Table Name**, you can edit these values later as well.

4. Click **Submit**.  
Now you can define the rest of the snapshot.
5. To specify the query that returns the data for the snapshot:
  - a. For **SQL Statement**, specify an SQL SELECT statement that retrieves the values you want in the snapshot. The query can refer to multiple tables. The query can use \* where that is syntactically valid.
  - b. Click **Parse SQL Statement**.

The system then tries to parse the SQL statement and determine the structure of the table it will generate and populate. If the system can parse the SQL, the **Columns** section then displays the fields of the new table, and the **Indices** section displays the indices it will generate for the new table.

If the system cannot parse the SQL, a warning is displayed, and you can edit the query.

6. Optionally add custom indices to this table. To add an index:
  - a. Click **New Index**.
  - b. For **Index Name**, type a unique index name.
  - c. For **Property 1**, select the field to index.  
To create an index on multiple fields, repeat with **Property 2** and so on.
7. If there is a table that you want to lock while performing this snapshot (in addition to the table or tables used in **SQL Statement**):
  - a. Click **Add Table**.
  - b. For **Table Name**, type the fully qualified table name (not just the short name).
  - c. Click **Submit**.
8. Click **Save**.

Notice that after you have saved a snapshot for the first time, the **Tables To Lock For Snapshot** section lists the table or tables to which **SQL Statement** refers.

**Tip:** After you run a newly defined snapshot, use the [SQL Explorer](#) to verify that the snapshot definition has captured the data you need to save.

## 11.4 See Also

- [Using the SQL Explorer](#)
- [Editing and Managing Snapshots](#)
- [Scheduling and Running Tasks](#)
- [Handling Task Errors](#)
- [Viewing Run History](#)





# 12

## Editing and Managing Snapshots (2.9)

Snapshots are static copies of data in InterSystems Data Fabric Studio™, for future review. Once you have defined snapshots, you can [run them](#), either on a schedule or manually. This page describes how to edit and manage them; [another page](#) describes the basics of defining them.

### 12.1 Viewing the Snapshots Dashboard

To view the Snapshots Dashboard, click the Snapshot  icon in the application menu.

This page enables you to see all the snapshots. For each snapshot, the page displays the following information.

- **Status**—Status of the snapshot, which can be **Idle**, **Running**, or **Error**.
- **Snapshot Name**—Name of the snapshot.
- **Snapshot Description**—Description of the snapshot.
- **Table Name**—Name of the table in which the current version of this snapshot is stored.
- **Life Cycle**—Displays the life cycle state of the snapshot, which can be **Active** (the initial state), **Disabled**, or **Deleted**.


Only **Active** snapshots can be [scheduled](#).

Click any snapshot in order to view it or make changes.

To filter the display, use the options above the column headers.

### 12.2 Editing a Snapshot

To edit a snapshot:

1. Click the Snapshot  icon in the application menu.
2. Click the snapshot you want to edit.  
The snapshot definition is displayed.
3. Click **Edit**.
4. Make changes in the same way as when you [create](#) a snapshot.



If you change the SQL query used in the snapshot, click **Parse Query** so that the system can update the structure of the associated [table](#), if needed. If the snapshot definition now defines a different set of columns, the system automatically generates a new table that has the required columns. The older table is left alone. The system displays an informational message about this change.

5. Click **Save**.

Or click **Delete Draft** to cancel your edits.


## 12.3 Retagging a Snapshot Run

If you have [run](#) a snapshot, you can change that tag that is associated with any particular run of that snapshot. To do so:

1. Click the Snapshot  icon in the application menu.
2. Click the snapshot.  
The snapshot definition is displayed.
3. Click **Snapshot History**.
4. In the row for the run that you want to retag, click the Edit  icon in the **Tag Name** column.
5. Type a new tag and press **Enter**.


## 12.4 Disabling a Snapshot

To disable a snapshot so that it cannot be run:

1. Click the Snapshot  icon in the application menu.
2. Click the snapshot you want to disable.  
The snapshot definition is displayed.
3. Click **Disable**.
4. Click **Disable** to confirm.


## 12.5 Enabling a Snapshot

To enable a previously disabled snapshot:

1. Click the Snapshot  icon in the application menu.
2. Click the snapshot you want to disable.  
The snapshot definition is displayed.
3. Click **Enable**.

## 12.6 Deleting a Snapshot

To delete a snapshot:

1. Click the Snapshot  icon in the application menu.
2. Click the snapshot you want to delete.  
The snapshot definition is displayed.
3. Click **Delete**.
4. Click **Delete** to confirm.

## 12.7 See Also

- [Defining Snapshots](#)
- [Scheduling and Running Tasks](#)
- [Viewing Run History](#)




# 13

## Managing Table Indices (2.9)

This page explains how to add custom indices to tables in InterSystems Data Fabric Studio™ and explains how the system automatically builds indices as needed.

### 13.1 Adding an Index to a Staging Table


Within the definition of a recipe, you can add custom indices to the generated staging table. To do so:

1. Click the Recipes  icon in the application menu.
2. If the recipe is in a group, expand that group.
3. Click the recipe name.
4. At the bottom of the page, see the list in **Indices and Staging Table Details**.  
This section shows one row for each data source used in the recipe.
5. Click the row for a given data source. The page then lists the generated staging tables associated with this data source.
6. Click a table name in this list. The right side of the page then provides information about the indices on the table, along with details about the fields in this table.
7. For **Index Name**, type a unique index name.
8. For **Property 1**, select the field to index.  
To create an index on multiple fields, repeat with **Property 2** and so on.
9. Because there is now a new index, the **Build Indices** button is enabled. You can click that to immediately start building this index for any existing data in this staging table.

However, even if you do not click **Build Indices**, the system automatically schedules a background task that will build this index for any existing data in this staging table. The system will also automatically index new data.

### 13.2 Adding an Index to a Snapshot

Within the definition of a snapshot, you can add custom indices to the generated snapshot table. To do so:

1. Click the Snapshot  icon in the application menu.
2. Click the snapshot you want to edit.
3. Click **Edit**.
4. Click **New Index**.
5. For **Index Name**, type a unique index name.
6. For **Property 1**, select the field to index.

To create an index on multiple fields, repeat with **Property 2** and so on.

When you add an index, the system automatically detects this change and schedules a background task that will build this index for any existing data in this table. The system will also automatically index new data.

## 13.3 Index Build Pending Message

Sometimes the process of building an index can be time-consuming and it is necessary to wait before performing recipe runs or snapshots that use the table being indexed. In such a case, the system displays a message indicating that it is in the process of building a required index. The system prevents you from prematurely using the data.

## 13.4 See Also

- [Editing and Managing Snapshots](#)
- [Editing and Managing Recipes](#)

# 14

## Defining Entities and Calendars (2.9)


A key part of InterSystems Data Fabric Studio™ is the ability to [schedule](#) automated processing, and these schedules generally need to align with relevant calendars. For example, a report may need to get run on every business day but not on holidays or weekends. However, not all regions and not all organizations necessarily use the same calendar.

To simplify the handling of calendars, the system provides the concept of entities. An *entity* defines a calendar; specifically, each entity has a time zone, a fiscal calendar, a business week, and a holiday schedule. An entity can be a child of another entity and can inherit some or all of that data from its parent.

In practice, an entity can represent a region, a subsidiary, or an organization. In any case, the entity defines a set of calendar information that you want to rely on, for scheduling purposes.

### 14.1 Defining an Entity


To create a top-level entity:

1. Click the Entity Master  icon in the application menu.
2. Click **Add Entity**.  
The system displays a dialog box where you can specify the details.
3. Enter the following information:
  - **Enable Entity**—Select this if you want the entity to be available for use in [scheduling](#).
  - **Entity Description**—Enter the name of this entity, which must be unique.
  - **Time Zone**—Select the time zone to which this entity belongs.
  - **Fiscal Year Start Month**—Select the first month of the fiscal year for this entity.
  - **Fiscal Year End Month**—Select the last month of the fiscal year for this entity.
  - **Business Week First Day**—Select the first day of the business week for this entity.
  - **Business Week Last Day**—Select the last day of the business week for this entity.
4. Click **Submit**.

Now you can [define the holiday calendar](#) for this entity. You can also [define child entities](#) that use parts of this entity's calendar data.

## 14.2 Defining a Child Entity

To create a child entity:


1. Click the Entity Master  icon in the application menu.
2. Select the entity to which you want to add a child.
3. Click **Add Child Entity**.

The system displays a dialog box where you can specify the details.

4. Enter the following information:
  - **Enable Entity**—Select this if you want the entity to be available for use in [scheduling](#).  
If the parent entity is not enabled, this entity cannot be enabled.
  - **Entity Description**—Enter the name of this entity, which must be unique within the parent entity.
  - **Use Parent Time Zone?**—Select this if you want the entity to use the time zone defined by the immediate parent entity.
  - **Time Zone**—Select the time zone to which this entity belongs. (Not available if **Use Parent Time Zone** is selected.)
  - **Use Parent Fiscal Calendar?**—Select this if you want the entity to use the fiscal year start and end month defined by the immediate parent entity.
  - **Fiscal Year Start Month**—Select the first month of the fiscal year for this entity. (Not available if **Use Parent Fiscal Calendar** is selected.)
  - **Fiscal Year End Month**—Select the last month of the fiscal year for this entity. (Not available if **Use Parent Fiscal Calendar** is selected.)
  - **Use Parent Business Week?**—Select this if you want the entity to use the business week start and end days defined by the immediate parent entity.
  - **Business Week First Day**—Select the first day of the business week for this entity. (Not available if **Use Parent Business Week?** is selected.)
  - **Business Week Last Day**—Select the last day of the business week for this entity. (Not available if **Use Parent Business Week?** is selected.)
  - **Use Parent Holiday Calendar?**—Select this if you want the entity to use the holiday calendar of the immediate parent entity. Note that you can [override the calendar details](#).
5. Click **Submit**.

## 14.3 Editing an Entity

To edit an entity:

1. Click the Entity Master  icon in the application menu.
2. Select the entity you want to edit.  
The entity definition is displayed, along with its [holiday calendar](#).




3. Make changes as needed, modifying the values previously described for [top-level entities](#) and [child entities](#).
4. Click **Save**.

**Note:** Changes to an entity do not affect any tasks that are currently running, because the entity for a task is checked at the start of the task run.

## 14.4 Editing a Holiday Calendar

To edit the holiday calendar for an entity:

1. Click the Entity Master  icon in the application menu.
2. Select the entity (first expanding the entity tree if necessary).

The system then shows the applicable holiday calendar, next to the entity details. If the selected entity is a child entity that inherits calendar information from its parent, the **Status** column displays **Inherited** for each inherited holiday.

Here you can do the following:

- Filter the display of holidays. To do so, select a year from the dropdown list above the table of dates. Or type the name (or partial name) of a holiday into the box above the table of dates. In either case, the display is immediately filtered.
- Add an entry. To do so:
  1. Click **Add Holiday**.
  2. For **Holiday Description**, enter a short holiday name.
  3. For **Holiday Date**, select a date.
  4. Click **Submit**.
- Remove an entry. To do so:
  1. Click **Delete** in the row for that entry.
  2. Click **Remove Holiday** to confirm.

You cannot remove an inherited entry, but you can deactivate it; see the next item.


- Deactivate an inherited entry. To do so, click the slider in the first column of row for that entry, moving the slider to the off position. The **Status** column will then display **Deactivated** for this holiday.
- Reactivate a previously deactivated (inherited) entry. To do so, click the slider in the first column of row for that entry, moving the slider to the on position. The **Status** column will then display **Inherited** for this holiday.

Each of these changes is immediately saved as part of the entity definition.

**Note:** Calendar changes do not affect any tasks that are currently running, because the entity for a task is checked at the start of the task run.

## 14.5 Deleting an Entity

To delete an entity:

1. Click the Entity Master  icon in the application menu.
2. Select the entity (first expanding the entity tree if necessary).  
The entity definition is displayed.
3. Click **Delete**.
4. Click **Delete Entity** to confirm.

## 14.6 See Also

- [Scheduling and Running Tasks](#)

# 15

## Scheduling and Running Tasks (2.9)


To schedule recipes and other tasks, you use the Business Scheduler, which you can also use to run tasks manually. There are three types of tasks:

- **Recipe**, which runs a single [recipe](#)
- **AtScaleCube**, which rebuilds a [cube](#)
- **Snapshot**, which performs a [snapshot](#) run

Before you can schedule any task, you must define and enable at least one [entity](#).

### 15.1 Scheduling a Task

To schedule a task:

1. Click the Business Scheduler  icon in the application menu.  
The system displays a table that lists the currently defined schedule items; if needed, use the [filters](#) at the top of the table to narrow down what is shown.
2. Click **Create** in the upper right and then click **New Task**.  
The system displays a table that lists the recipes, snapshots, and cube builds that can be scheduled. If needed, use the filters at the top of the table to narrow down what is shown.
3. Click the row corresponding to the item to schedule.  
The system displays a page where you specify details about the item.
4. Specify the following information:
  - **Task Description**—*Required*. Type a brief description for this task.
  - **Entity**—*Required*. Select the [entity](#) whose calendar information you want to use. Note that this field is set automatically if you specify **Task Group** as a [schedule group](#).
  - **Task Group**—Optionally select the [task group](#) that this task should be in. If the task group is a schedule group, the system automatically sets **Entity** equal to the entity for that task group.
  - **Enabled**—Optionally clear this check box if you want to disable this task (for example, if you are not ready to run or schedule it).
  - **Tag**, **Dependency Expression**, and **Dependency Inactivity Timeout**—See [Managing Task Dependencies](#).

- **Workflow Role for Handling Exceptions**—*Required.* Select the [role](#) that should receive any [workflow items](#) in case of exceptions related to this task.
- **Email Distribution List for Error Notifications**—Select the [email distribution list](#) that should receive messages when this task encounters an error.
- **Email Template for Error Notifications**—Select the [email template list](#) to use when sending messages when this task encounters an error. This is required if you select **Email Distribution List for Error Notifications** and is hidden otherwise.
- **Email Distribution List for Success Notifications**—Select the [email distribution list](#) that should receive messages when this task runs successfully.
- **Email Template for Success Notifications**—Select the [email template list](#) to use when sending messages when this task run successfully. This is required if you select **Email Distribution List for Success Notifications** and is hidden otherwise.

In the **Scheduling Details** section, select **Run Mode**. The available values are as follows:

- **Manually Run**—The task is not run until a user chooses to run it.
- **Run on Schedule**—The task is run at the scheduled date and time. For this run mode, specify additional options to [control the scheduling](#).

If you specify a **Dependency Expression**, that dependency is considered. If it is the time for the task to run and that dependency expression is not satisfied, the task remains in the RunningWait state until a timeout occurs or the dependency is satisfied. That allows the task to wait for other tasks to run or to wait for external events that may indicate, for example, that data is ready to be retrieved.

This option is not available for a task within a [schedule group](#).

- **Run After**—The task is run only after another task has been successfully run.

For example, suppose we have a task B that should run right after another task A finishes running successfully. Task B is initially in the state RunningWait, meaning that it is waiting for task A to finish running. When task A successfully completes, task B moves into the Running state. When task B finishes running, it moves back to RunningWait.

- **Run After Dependency**—The task will be run every time the dependency is satisfied. For this run mode, **Dependency Expression** is required, and it must include at least one condition that uses the WHEN() function. For example, suppose task C must run every time an external business event with tag DataIsReady happens (and this can happen many times a day). We can configure task C to be of type **Run After Dependency** with a dependency expression WHEN(DataIsReady). This task will immediately go to RunningWait, waiting for the business event DataIsReady to happen. Once we receive such an external business event, the task will run and immediately move back to RunningWait status, waiting for the next DataIsReady to be issued.

With this mode, you may or may not need a timeout (**Dependency Inactivity Timeout**). A timeout is probably not important if the dependency expression is a simple WHEN(). But if the dependency expression contains a TODAY() condition *and* that event never comes, a timeout would allow this task to fail and start a workflow that will let users know that no tag for that TODAY() has arrived yet.

- **Run With Group**—The task will be run when the parent [schedule group](#) is run.

This option is available only to a task within a schedule group.

## 5. Click **Save Task**.

## 15.2 Scheduling Details

If the **Run Mode** for a task is **Run on Schedule**, the Business Scheduler displays additional options to control the scheduling. The options include the following:

- **Type of frequency to run at**—*Required*. Select the option that best describes how often to run this task.  
Depending on your selection, the page may show additional items to control the scheduling, such as the specific days of the week, the specific months, and so on.
- **On Holidays** —*Required*. Select the option that describes how to handle holidays: **Don't Run**, **Run Anyway**, **Run Next Business Day**, **Run Previous Business Day**.
- **Run on the Following Days of the Week** —*Required if task runs every calendar day or every business day*. Select each day on which the task should be run.
- **Start Time on Scheduled Days**—*Required*. Select the time of day when the task should be started (using the local time zone defined for the associated entity).
- **End Time on Scheduled Days**—*Required if task runs more than once a day*. Select the time of day after which the task should not be started (using the local time zone defined for the associated entity).
- **Schedule Task End Time**—Optionally select this if you want the task to stop running at a specific date and time. If you select this option, also specify **Task End Date** and **Task End Time**.

The system displays additional fields to prompt you for information as needed. For example, if **Type of frequency to run at** is **Run Every X Minutes**, the system displays the field **Run Every X Minutes**, which you use to specify the number of minutes.

## 15.3 Managing Task Dependencies

There may be specific tasks that you want to run only after other tasks have been run, on the same day; the system provides a way to manage these dependencies. To specify that one task (task A) should not be run until another task (task B) has completed, you use the following system:

- Decide on a unique, short tag that describes this dependency (for example, `backup`). Note that tags are case-sensitive.
- When scheduling the task or tasks that need to occur first (in our example, task B), specify the **Tag** field, using the tag you have decided on.
- When scheduling the dependent task (task A), specify **Dependency Expression**, which specifies the condition (or combination of conditions) that must be met in order for this task to be started.

In its simplest form, a dependency expression has the form `FUNCTIONNAME(tag)`, where `FUNCTIONNAME` is `TODAY`, `WHEN`, or `WINDOW` and `tag` is a tag that has been applied to some other task. The meaning of the expression depends on the [function](#). As an example, consider the following example dependency expression:

```
TODAY(backup)
```

This expression means that the task should not be started until the completion (today) of a task that is tagged with the `backup` tag.

You can combine dependency expressions via the keywords `AND` and `OR`, along with parentheses. For example:

```
TODAY(tagA) OR (TODAY(tagB) AND TODAY(tagC))
```

- In addition to specifying **Dependency Expression**, you can specify **Dependency Inactivity Timeout** in seconds.

## 15.4 Functions Available in Dependency Expressions

This section provides reference information on the functions that can be used in [dependency expressions](#):

### TODAY(tag)

Where *tag* is a tag of the event to wait for. This dependency expression means to wait until the completion (today) of a task that is tagged with *tag*.

Unlike the WINDOW() function, TODAY() considers only events within the same day.

### WHEN(tag)

Where *tag* is a tag of the event to wait for. This dependency expression means to wait until the completion of a task that is tagged with *tag*. When a task starts running and it has the WHEN(SomeTag) function in its dependency expression, the system will hold the task in the RunningWait status until a business event identified by SomeTag arrives. This business event must arrive/happen *after* the task reaches the RunningWait status. When such a business event happens, the task will then proceed to Running unless the dependency expression has other conditions that must be satisfied.

If the task takes too long to run and return to its RunningWait status, and a new business event expected by the WHEN() function arrives while the task is still running, that event will be lost since it did not arrive *after* the task is back in RunningWait.

### WINDOW(tag,lower,higher)

Where:

- *tag* is a tag of the event to wait for.
- *lower* is a positive number of hours. This argument specifies how old the event can be, from the moment the system began checking for it.
- *higher* is a positive number of hours. This argument specifies how long to wait for the event, from the moment when the system began checking for it.


This dependency expression means the system will either run the task if the given event was completed within the last *lower* hours ago or will wait *higher* hours for that event to complete. This window of time can span days. For example, consider a task (called Sample Task) that uses this dependency expression:

```
WINDOW(backup, 4, 4)
```

Suppose that Sample Task starts running at 11 PM. In this case, the system first checks to see if the backup task has completed within the last four hours (that is, after 7 PM); if so, the system starts Sample Task immediately. On the other hand, if the backup task has not completed yet, the system waits, potentially until 3 AM. At any time before 3 AM, if the backup task is completed, the system starts Sample Task after the backup task is completed.

## 15.5 Modifying a Task


To modify a task definition (for example, to change the schedule details):

1. Click the Business Scheduler  icon in the application menu.  
The system displays a table that lists the items that are currently scheduled. If needed, use the filters at the top of the table to narrow down what is shown.
2. In the row for the task, right-click and select **Edit Task**.
3. Modify details here in the same way as when you [schedule](#) a task.
4. Click **Save Task**.  
The system runs the task (or attempts to) within the next minute. When it does so, it updates the **Run History** section of this page.

Also see [Managing Task Groups](#) for information on moving tasks into and out of groups.


## 15.6 Running a Task Manually

To run a task manually:

1. Click the Business Scheduler  icon in the application menu.  
The system displays a table that lists the items that are currently scheduled. If needed, use the filters at the top of the table to narrow down what is shown.  
Tasks that can be run manually are displayed here with **Next Run** as **Manual**; these are tasks [defined](#) with the **Manually Run** check box selected.
2. In the row for the task, right-click and select **View Task**.
3. Click **Manually Run Now**.
4. Click **Run Task** to confirm.  
The system runs the task (or attempts to) within the next minute. When it does so, it updates the **Run History** section of this page.

## 15.7 Aborting a Task

To stop a task that is currently running:


1. Click the Business Scheduler  icon in the application menu.
2. In the row for the task, right-click and select **View Task**.
3. Click **Abort Task**.
4. Click **Abort Task** to confirm.

The system runs the task (or attempts to) within the next minute. When it does so, it updates the **Run History** section of this page.

Also see [Handling Task Errors](#).

## 15.8 Deleting a Task

To delete a task definition:

1. Click the Business Scheduler  icon in the application menu.
2. In the row for the task, right-click and select **View Task**.
3. Click **Delete**.
4. Click **Delete** to confirm.

## 15.9 See Also

- [Defining Entities and Calendars](#)
- [Defining Recipes](#)
- [Defining Snapshots](#)
- [Defining Cubes](#)
- [Filtering and Customizing the Business Scheduler](#)
- [Handling Task Errors](#)



# 16

## Managing Task Groups (2.9)

The [Business Scheduler](#) provides an option to organize tasks into groups. There are two types of task groups:


- Ordinary groups. These groups cannot be scheduled; their only purpose is to visually organize the Business Scheduler Dashboard.
- Schedule groups. These groups can be scheduled.

For each task group, the Business Scheduler summarizes the statuses of all the tasks in the group; by design, this is the most important status of the group. For instance, if all tasks in a group are Running, but one task is in Error, the group summary status is Error. This way, even when a group is collapsed, the Business Scheduler displays the status that most needs attention.

**Note:** In addition to the options described on this page, you can move tasks into and out of groups by [editing the tasks](#) as you do with other kinds of changes to the tasks.


### 16.1 Creating an Ordinary Group

To create an ordinary group:

1. Click the Business Scheduler  icon in the application menu.
2. Click **Create** in the upper right and then click **Group**.
3. Enter a value for **Task Group Name**.
4. Click **Submit**.

### 16.2 Creating a Schedule Group

To create a schedule group:


1. Click the Business Scheduler  icon in the application menu.
2. Click **Create** in the upper right and then click **Scheduled Group**.

The system then displays a page where you specify the schedule details. This page requires the same information as when you schedule a task. See [Scheduling a Task](#).

3. Click **Save Task**.

## 16.3 Moving a Task into a Group


To move a task into a group, you can move the cursor onto the selector at the start of the row for that task and then drag and drop to the row for a task group. Or you can do the following:

1. Click the Business Scheduler  icon in the application menu.
2. Right-click in the row for the task that you want to move.
3. Click **Change Group**.
4. Select an existing **Task Group**.
5. Click **Submit**.

If you move a task into a schedule group, the entity governing the scheduling for that task is reset to equal the entity that governs the schedule group. For example, suppose that Task A uses the entity US and Schedule Group 100 uses the entity EU. If you move Task A into Schedule Group 100, the entity for Task A becomes EU. Similarly, the schedule option **Run On Schedule** is not supported for tasks within a schedule group. If Task A uses **Run On Schedule**, when you move that task into Schedule Group 100, its schedule option is changed to **Run with Group**.


## 16.4 Moving a Task out of a Group

To move a task out of a group (but not into a new group), you can move the cursor onto the selector at the start of the row for that task and then drag and drop to bottom of the table. Or you can do the following:

1. Click the Business Scheduler  icon in the application menu.
2. Right-click in the row for the task that you want to move.
3. Click **Ungroup**.

## 16.5 Renaming a Task Group


To rename a task group:

1. Click the Business Scheduler  icon in the application menu.
2. Right-click in the row for the task group that you want to rename.
3. Click **Rename Group**.
4. Enter a new value for **Task Group Name**.

5. Click **Submit**.

## 16.6 Deleting a Task Group

To delete a task group:

1. Click the Business Scheduler  icon in the application menu.
2. Move all tasks out of the group that you want to delete.
3. Right-click in the row for the task group.
4. Click **Delete Group**.

## 16.7 See Also

- [Scheduling and Running Tasks](#)
- [Filtering and Customizing the Business Scheduler](#)



# 17

## Handling Task Errors (Using the Workflow Inbox) (2.9)

This page describes what happens and how to proceed when an error occurs during a recipe, snapshot run, or other task.


### 17.1 When the System Encounters an Error

When the system encounters an error while executing a task (either scheduled or manually run), the system does several things:

- Sends email to a specified email distribution list (as specified in the [task schedule](#)).
- (In most cases) Generates a workflow task, and assigns the task to the applicable workflow role (as specified in the [task schedule](#)).
- Updates the display within Business Scheduler to indicate the error condition.
- Updates the display of the associated recipe or snapshot to indicate the error condition.


### 17.2 First Steps

The first step is to claim ownership of the problem and determine its cause, as follows:

1. Click the Workflow Inbox  icon.
2. Click **Unassigned**.  
The system displays a list of the unassigned tasks.
3. Click the task to display its details.
4. Click **Accept**. This assigns the task to you.
5. Optionally type a note into the **Comment** field and click **Save**.
6. Then see [Addressing a Task](#).

## 17.3 Viewing the Workflow Inbox

The Workflow Inbox shows all the workflow tasks. To display it:

1. Click the Workflow Inbox  icon in the application menu. The system initially displays all tasks assigned to you, within the searchable task list.
2. Optionally click any of the following options, to display a different set of tasks:
  - **All** displays all tasks, for all roles, showing both incomplete tasks and completed tasks.
  - **My Tasks** displays tasks assigned to any of the roles that you belong to.
  - **Unassigned** displays any unassigned tasks.
  - **Completed** displays all completed tasks.
  - **My Completed** displays all completed tasks assigned to any of the roles that you belong to.
3. Optionally type into the search bar above the task list. The system then displays only the tasks whose text contains a match for your search string.

When you click a task, the system displays details for that task.

## 17.4 Addressing a Task

To address a workflow task, [first](#) assign the task to yourself. Read the message to determine what the error is and how you plan to correct it. For example, you may need to do any of the following:

- Ensure that the file to be loaded has the correct filename pattern and exists in the correct location (in the case of recipes involving a file-based data source).
- Correct the data being loaded (for example, if a value is out of range or is the wrong type). This may involve making a correction in a source system.
- Reimport a schema, if the new data does not match the schema in the system.
- Adjust a schema (changing data types or required fields, for example).
- Adjust a step in a recipe.
- Adjust the query used by a snapshot.

Then, depending on the kind of underlying activity, you may have some or all of the following options within the workflow task:

- **Abort**—Halts the task without retrying it.  
Once you have aborted a task, you cannot retry it.
- **Retry**—Retries the task. This is suitable if the data has been corrected or if a missing file has been made available. It is not useful if you have made changes to a schema or to a recipe.
- **Retry All with New Version**—Retries the task with the new version of the recipe, considering new versions of the failed activity and of any later activities. The system will retry the recipe from the point where it failed, using the updated version of that activity and updated versions of any subsequent activities.

- **Retry Failed Activity with New Version**—Retries the task with the new version of the recipe, considering only the new version of one activity in the recipe. The system assumes that the new activity is the one where the failure occurred, and it will retry the recipe from the point where it failed, using the updated version of that one activity. It uses the original version of any other activities in the recipe.

## 17.5 Relinquishing a Task

If a task is assigned to you and another person should address the task instead of you, you should relinquish the task so that the other person can accept it. To relinquish a task:

1. Display the task.
2. Click **Relinquish**.

The task immediately becomes unassigned. If the task had a comment, the comment is removed. Now another person can accept the task.

## 17.6 See Also

- [Scheduling and Running Tasks](#)





# 18

## Viewing Run History (2.9)


The Business Scheduler displays the [run history](#) and current [status](#) for all tasks. The *run history* consists of information such as when the task was run, who ran it, how long it took, and so on.


For recipes, the run history also provides a link to a report (the *run report*) that has details on dropped records, validation errors, and reconciliation errors.

For snapshot, you can also view run history from the [snapshot definition](#).

### 18.1 Viewing Run History

You can view the run history of any task from within the Business Scheduler. To do so:

1. Click the Business Scheduler  icon in the application menu.  
The system displays a table that lists the currently defined tasks
2. Optionally use the [filters](#) at the top of the table to narrow down what is shown.

By default, this page lists all tasks and task groups. The task rows include a selector icon  at the left side; the task group rows do not have this icon.

This page displays the following columns:

- **Task Description**—For a task, this displays the short name of the task (generally the name of the [recipe](#) or [snapshot](#) that the task is based on).  
For a task group, this column displays the group name in bold with an icon to collapse or expand the group. If the group is empty, the table displays (empty) after its name and there is no option to expand the group.
- **Status**—For a task, this indicates the [status of the task](#). For a task group, this displays a summary status, which is intended to be the most important status of the tasks in the group — the status that most needs attention. For instance, if all tasks in a group are Running, but one task is in Error, the group summary status is Error. This way, even when a group is collapsed, the Business Scheduler displays the information that you need to see.
- **Type**—Indicates the row type: **Recipe**, **Snapshot**, **AtScaleCube**, or **Group**.
- **Resource Name**—Indicates the resource to which this task applies.

This column is empty for task groups, as are all the remaining columns.

- **Exceptions**—If the system encountered an error during its most recent task run, this field displays an error message and includes a link to the [Workflow Inbox](#), where you can see the details.  
See [Handling Task Errors](#).
- **Entity**—The [entity](#) whose schedule this task uses.
- **Enabled**—Indicates whether this task is currently enabled.
- **Last Run Report**—A link to the most recent [run report](#) for the task (available only for recipes).
- **Task Start Time**—When exactly this task started running.
- **Task Wait Time**—The amount of time spent on waiting for dependencies such as waiting for an event to arrive or waiting for a previous run of the job to finish.
- **Job Start Time**—When the job (recipe, snapshot, cube) actually started running.
- **Job Run Time**—How long did it take for the job to finish running.
- **Job End Time**—When did the job finish running.
- **Total Elapsed Time**—How long did the entire task take to run (including waiting time).
- **Next Run**—Indicates when or how the task will next be run. This is either a future date and time or is simply the word **Manual** (which means that this task is defined to be run manually).
- **Tag**—The [tag](#) applied to this task.
- **Dependencies**—The [dependency expression](#) for this task, if any.


## 18.2 Task Status

The status of any task can be any of the following:

- **Idle**—The task is not running.
- **Starting**—The task is starting.
- **RunningWait**—The task is running but is in a waiting state.
- **Running**—The task is actively running.
- **Error**—The task has encountered an error and has stopped. See [Handling Task Errors](#).
- **Complete**—The task has completed.
- **Aborted**—A user [aborted](#) the task (or the system automatically aborted the task).

## 18.3 Downloadable Run Reports (Recipes)

To display the run report for a recipe:

1. Click the Business Scheduler  icon in the application menu.  
The system displays a table that lists the currently defined tasks
2. Click the **Last Run Report** link for the recipe.

This displays a page that reports on all activities within the recipe. The page has expandable sections, by activity type, so that you can display only the data of interest to you.

When you view the validation and reconciliation sections, you have the option of downloading the validation and reconciliation reports, respectively.

## 18.4 See Also

- [Handling Task Errors](#)
- [Scheduling and Running Tasks](#)
- [Purging Old Recipe Data](#)



# 19

## Filtering and Customizing the Business Scheduler (2.9)

The Business Scheduler displays all tasks and [task groups](#). Depending on the number of items defined in the [Business Scheduler](#), you may find it helpful to filter the display, to customize the display, or both.

### 19.1 Filtering the Display


To filter the display, use the dropdown menus and type-in boxes above the table.


To clear all filters, click the grid  icon and then click **Clear Filters**.

### 19.2 Customizing the Columns

You can customize the columns that the Business Scheduler displays. This customization does not affect other users, and this customization is preserved until you modify it or restore the default set of columns.

To customize the columns:

1. Click the grid  icon and then click **Toggle Columns**.  
This displays a list of the columns that can be hidden or shown.
2. Clear or select check boxes in this list, depending on which columns you want to hide or show.
3. Click anywhere outside of the list to dismiss the list.

To restore the default set of columns, click the grid  icon and then click **Reset Columns**.

### 19.3 See Also

- [Viewing Run History](#) (provides details on the columns shown in the Business Scheduler)
- [Scheduling and Running Tasks](#)

- [Managing Task Groups](#)
- [Handling Task Errors](#)

# 20

## Using the File Manager (2.9)

The File Manager enables you to manage data files and schema files within the InterSystems Data Fabric Studio™ file system. Specifically, each file-based [data source](#) defines and uses an associated set of directories within this file system. Via the File Manager, you can view the contents of those directories (other than the actual file contents), remove files, and upload new files.

Typically you use the File Manager to upload schemas and to upload data for test purposes. In production use, the recommendation is to create an automation that writes files to the applicable directories from the source system or systems. As a consequence, you might use the File Manager only during the early phases of your work with the system.

### 20.1 Standard Subdirectories for a File Source

Each file-based [data source](#) defines and uses its own directory within the file system, given the name that you provide when defining that data source. That directory includes the following standard subdirectories:

- **Samples**—The purpose of this subdirectory is to hold any sample files that define schemas. You can upload samples to this directory and then import them into the [Data Catalog](#).  
These files remain in this directory until you delete them, if ever.
- **Source**—This is the input directory for the data source. When you run a [recipe](#) that loads data from this data source, the recipe looks for a file in this directory (and the file name must match the filename pattern specified by the associated schema; see [Editing a Schema](#)).


You can upload files to this directory or you can create an automation that writes files here from the appropriate source. You can manually delete files from this directory.

- **Work**—When a recipe is started, the input file is moved from **Source** to **Work**. This prevents the file from being accidentally processed by a staging activity in a different recipe.
- **Archive**—When a file has been successfully processed, it is moved from **Work** to **Archive** and is renamed with the timestamp appended to the original name.

(If a file is not successfully processed, it is moved back to the Source directory.)

### 20.2 Viewing the Directory Contents for a File Source

To view the directories associated with a file-based data source:



1. Click the File Manager  icon in the application menu.
2. For **Choose a File Directory Source**, select the data source of interest.

The system then displays a list of folders like this:

List of Folders in the excel1 File Source	
Folder Name	
/Archive	
/Samples	
/Source	
/Work	

The folder names are always Archive, Samples, Source, and Work.


3. Click any folder name to display the current contents of that folder. For example:

List of Files in the Samples Folder			
		<input type="text" value="Search by file name..."/>	<input type="button" value="Upload Files"/>
File Name	File Size	Date Uploaded	Actions
sampledata.xlsx	9.72 KB	8/23/2023, 12:17:57 PM	
comparisondata.xlsx	9.72 KB	8/23/2023, 12:17:39 PM	

4. Optionally edit **Max Number of Files to View**. This change immediately affects the display.

## 20.3 Uploading Files

To upload one or more files:

1. Click the File Manager  icon in the application menu.
2. For **Choose a File Directory Source**, select the data source of interest.
3. Select either the **Samples** or the **Source** directory, depending on where you want to load this file.

The page then displays any existing files in that directory.

4. Click **Upload Files**.

This displays a new area on the right, the *upload area*.

5. Perform the following steps within the upload area:


- a. Drag and drop one or more files from your local computer to the upload area.

Or click the upload area and then browse to the directory that contains the files. Then select one or more files.

As you do this, the upload area displays a list of files that can be uploaded.

Note that you cannot upload a file larger than 500 MB.





- b. Optionally, if you have selected files that you do not want to upload, click the Remove  icon next to those files. Or click **Clear Files**.
- c. Click **Upload File(s)** in the upload area.

Now the files are uploaded and are displayed in the list of contents for the given directory. The upload area is reset and shows no files.

Note that the new files overwrite any existing files that have the same name. The upload process does not rename files. Also when you upload a new copy of a schema file, that does not automatically reimport the schema. When you are ready to reimport the schema, do that as described in [Reimporting a Schema](#).

## 20.4 Deleting a File

To delete a file:

1. Click the File Manager  icon in the application menu.
2. For **Choose a File Directory Source**, select the data source of interest.
3. Select a directory name.
4. Click the Delete  icon next to the file.
5. Click **Remove** to confirm the action.

## 20.5 See Also

- [Importing Schemas](#)
- [Defining Recipes](#)




# 21

## Exporting and Importing Configurations (2.9)

For your solution, InterSystems (or other solution partner) sets up and maintains multiple parallel [environments](#) so that development and testing activities can safely occur without disturbing mission-critical activities. To simplify the process of keeping the environments synchronized, InterSystems Data Fabric Studio™ provides options for exporting and importing configurations. Here the term *configuration* refers to any of the items that you configure within the solution, including schemas, recipes, users, and roles.

### 21.1 Exporting Configurations

To export one or more configurations:

1. Click the Configuration Management  icon in the application menu.
2. Click **Export Configurations**.
3. Click the check box next to each configuration that you want to export.
4. Click **Export Selected**.

The system creates a ZIP file that contains all the selected configurations and it downloads that file to the download location specified by the browser. The name of the file has the form `BundleZip<datetime>.zip`.

The ZIP file contains one file for each configuration, and for those files, the filename format is *itemname.itemtype*, for example: `MyTestRecipe.TotalViewRecipe`.

### 21.2 Editing Configurations


You can unzip, edit, and rezip a configuration file. Do not change the name of the file.

Use the following command to rezip the file:

```
ditto -ck --rsrc --sequesterRsrc bundleFolder bundle.zip
```

## 21.3 Importing Configurations

To import one or more configurations:

1. Create a ZIP file that contains the configuration files.
2. Click the Configuration Management  icon in the application menu.
3. Click **Import Configurations**.
4. Click in the upload area and navigate to your file.
5. Click **Import All**.

The system displays a dialog box indicating whether the import was successful, along with information about any errors.

6. Click **Close**.

## 21.4 See Also

- [About Your Solution](#)