

TIBCO Cloud™ Integration - Flogo® (PAYG) Activities and Triggers Guide

Version 2.11.0

January 2021



Contents

TIBCO Documentation and Support Services	5
Out-of-the-box Activities and Triggers	6
General Category	6
Triggers	6
Timer Trigger	7
REST Trigger - ReceiveHTTPMessage	7
GraphQL Trigger	12
gRPC Trigger	13
Activities	14
ConfigureHTTPResponse	15
GRPCActivity	16
InvokeLambdaFunction	17
InvokeRESTService	18
Using SSL	22
LogMessage	22
Mapper	23
ParseJSONActivity	24
ReplyToHTTPMessage (Supported for backward compatibility only)	25
SendMail	25
Sleep	27
Apache Avro	28
Overview	28
Avro Serialize	28
Avro Deserialize	29
Apache Kafka	29
Overview	29
Configuring a Kafka Client Connection	29
Kafka Client Configuration Details	30
Kafka Consumer Trigger	32
Kafka Producer	34
Kafka Offset Commit	37
Apache Pulsar	37
Overview	37
Creating an Apache Pulsar Connection	38
Apache Pulsar Connection Details	38
Apache Pulsar Consumer Trigger	39
Apache Pulsar Producer Activity	41

Microsoft SQL Server	43
Overview	43
Creating a Microsoft SQL Server Connection	43
Microsoft SQL Server Connection Details	43
SQLServer Query	45
SQLServer Insert	47
SQLServer Update	49
SQLServer Delete	50
Manually Configuring Metadata	51
MQTT	51
Overview	51
Creating the MQTT Connection	51
MQTT Connection Details	52
Rules for Application Property	52
MQTT Subscriber Trigger	53
MQTT Publish Activity	55
Oracle MySQL	56
Overview	56
Creating an Oracle MySQL Connection	56
Oracle MySQL Connection Details	57
TLS Modes	58
MySQL Query	58
MySQL Insert	60
MySQL Update	62
MySQL Delete	63
Manually Configuring Metadata	64
PostgreSQL	65
Overview	65
Creating a PostgreSQL Connection	65
PostgreSQL Connection Details	65
TLS Modes	67
PostgreSQL Query	67
PostgreSQL Insert	69
PostgreSQL Update	71
PostgreSQL Delete	72
Manually Configuring Metadata	73
TIBCO Cloud Messaging	73
Overview	74
Creating a TIBCO Cloud Messaging Connection	74

TIBCO Cloud Messaging Connection Details 74

MessageSubscriber Trigger 75

TCMMessagePublisher 77

TCMMessageAck 78

Legal and Third-Party Notices 79

TIBCO Documentation and Support Services

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product. To access the latest documentation, visit <https://docs.tibco.com>.

Product-Specific Documentation

Documentation for TIBCO Cloud™ Integration - Flogo® (PAYG) is available on the [TIBCO Cloud Integration - Flogo \(PAYG\) Product Documentation page](#).

The following documents can be found on the TIBCO Documentation site:

- *TIBCO Cloud™ Integration - Flogo® (PAYG) Release Notes*
- *TIBCO Cloud™ Integration - Flogo® (PAYG) Getting Started*
- *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide*
- *TIBCO Cloud™ Integration - Flogo® (PAYG) Activities and Triggers Guide*

How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

- For an overview of TIBCO Support, visit <http://www.tibco.com/services/support>.
- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support portal at <https://support.tibco.com>.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to <https://support.tibco.com>. If you do not have a user name, you can request one by clicking Register on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the [TIBCO Ideas Portal](#). For a free registration, go to <https://community.tibco.com>.

Out-of-the-box Activities and Triggers

This section documents the out-of-the-box activities and triggers that are supported in TIBCO Cloud™ Integration - Flogo® (PAYG).

General Category

The **General** category is available by default in all flows. It consists of activities and triggers that may be commonly used by any flow in the app. A trigger activates the flow in which it appears. An activity is used to perform a task.

This section contains the following topics:

Triggers

In addition to the **Receive HTTP Message** and **Timer Trigger** available for general use, TIBCO Cloud Integration - Flogo (PAYG) supports triggers that were originally created in Project Flogo™. This allows for a seamless import of apps that were created in Project Flogo™. The Project Flogo™ triggers are marked with the OSS abbreviation on them.

If you are creating an app in TIBCO Cloud Integration - Flogo (PAYG), it is preferable to use the general purpose triggers (the triggers that do not have an OSS tag on them), since they are more robust in functionality.

Refer to <https://github.com/project-flogo/contrib> for details on the activities that are marked with an OSS tag.

Trigger configuration fields are grouped into Trigger Settings and Handler Settings. A single trigger can be associated with multiple handlers.

- **Trigger Settings** - these settings are common to the trigger across all flows that use that trigger. If and when a flow attached to the trigger changes any **Trigger Settings** field, the change gets propagated to all flows attached to the trigger. A warning message gets displayed saying so and asking you to confirm before the changes are committed.
- **Handler Settings** - these settings are applicable to a specific flow attached to the trigger. Hence, each flow can set its own values for the **Handler Settings** fields in the trigger. To do so, open the flow and click on the trigger to open its configuration dialog. Click the **Settings** tab and edit the fields in the **Handler Settings** section.



You cannot create a flow branch from a trigger.



You can create the trigger at the time of flow creation or create a blank flow to begin with and attach the flow to one or more triggers at a later time after the flow has been created. If you anticipate that you will need to attach the flow to multiple triggers, be sure to create a blank flow and attach it to the triggers as needed. If you attach a flow to a trigger during flow creation, you cannot modify the trigger settings from within the flow. Refer to the section, "Selecting a Trigger When Creating a New Flow" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for more details.

For triggers that have an output, the output from the trigger becomes the input to the flow. Likewise, the output from the flow becomes the reply from the trigger.

This section contains the following topics:

Timer Trigger

Use the **Timer Trigger** trigger as a process starter when creating flows designed to be activated without external input. It is useful when you want your flows to run at certain time intervals. You can configure the **Timer Trigger** trigger to activate the flow multiple times at a specified interval.

Field	Description
Handler Settings	
Repeating	<p>True: Select the True radio button to run the flow at periodic intervals.</p> <p>False: Select the False radio button if you want the flow to run only once.</p> <p>If Repeating is set to True and a Time Interval and Interval Unit is specified, the flow will be triggered at the exact same time as the first run. For example, if the flow was run at 1:00 pm on Monday and the Interval Unit is specified as week with the Time Interval as 2, the flow be run at 1:00 pm on every other Monday.</p>
Time Interval	An integer indicating the number of units specified in the Interval Unit field. For example, if you enter 1 as the Time Interval and select Hour in the Interval Unit drop down menu, the Timer trigger activates the flow every hour. If you enter 2 as the time interval and specify Week as the Interval Unit , the flow is run every other week.
Interval Unit	The unit of time to use with the Time Interval field to determine how often to run the flow. The units can be: Second, Minute, Hour, Day, and Week.

REST Trigger - ReceiveHTTPMessage

Use the **ReceiveHTTPMessage** REST trigger when creating flows that require invoking RESTful web services that provide some input necessary to activate the flow. The **ReceiveHTTPMessage** trigger exposes your flow as an API, making it accessible by other apps running on either TIBCO Cloud™ or elsewhere. This trigger must be configured to set up the fields for a request that the server receives from a REST client.



If you make changes to the trigger such as add or delete path or query parameters, be sure to click the **Sync** button in order for the changes to be propagated to the flow input schema.

Trigger Settings

Field	Description
Trigger Settings	
Port	<p>By default, the trigger listens on the port 9999. You can change this to use another port that is open.</p> <p> If the app has multiple triggers that require a port to be specified, specify a unique port number for each trigger. Two triggers in the same app cannot run on the same port.</p>

Field	Description
Configure Using API Specs	<p>While creating a REST trigger, you can configure it by uploading an API specification file. If the API specification changes, you can merge the changes into an existing app by uploading the specification file again.</p> <p>To do this, click True and specify the following. (The default is False.)</p> <ul style="list-style-type: none"> • API Spec: Click Browse and then select the specification file to be used for configuring the trigger. Supported specifications are Swagger Specification 2.0 and OpenAPI Specification 3.0. • Path: All the paths from the specification file are listed in the drop-down list. Select a path. • Method: All the methods relevant to the selected path are listed in the drop-down list. Select a method. <p>If False is selected, you need to specify the Path and Method.</p>
Secure Connection	<p>By default, it is set to False. If you set this field to True, you can create a secure endpoint by providing Server Key and CA or Server Certificate.</p> <p>Server Key - A PEM encoded private key file</p> <p>CA or Server Certificate - A PEM encoded CA or self-signed server certificate file</p>
Handler Settings	

Field	Description
Path	<p>The resource path for the operation. By default, the path displayed here is the resource path you had entered when you created the flow. The Path field is editable. For example, if you want to add a path parameter for a GET operation, you can do so by editing the resource path in the GET flow. If you edit the path in the Path field for a particular REST operation flow, the edited resource path is applicable only to the flow in which it was edited.</p> <p>Two resource paths with same base path should not contain path parameters at the same location. For example, you cannot have the following paths in the same application:</p> <ul style="list-style-type: none"> • <code>/books/{ISBN}/Author/{releaseDate}</code> and <code>/books/{ISBN}/Author/releaseDate</code> is considered the same from a routing perspective. In these two paths, since the ISBN value is dynamic, it causes a conflict during path resolution. • <code>/books/{ISBN}/{releaseDate}</code> and <code>/books/{ISBN}/Author</code> in the same application is not supported. Although the two paths appear to be different, when a message comes in, the router mechanism cannot know which path to call (the one with parameter or the one without) since the actual value has been substituted for the parameter. • Resource path with two different path parameters at the same URL sub-section. For example, <code>/0.6/api/account/{account}/orderhistory/{orderhistory}/branch/{branch}</code> and <code>/0.6/api/account/{AccountKey}/Price?ProductList={ProductList}</code> In these paths even though the paths differ after the base path (<code>/0.6/api/account/</code>), there is a conflict when resolving the <code>{account}</code> and <code>{AccountKey}</code> values. • Multiple REST resources with same base path and same number of path parameters. For example, <code>/resource/{id}</code> and <code>/resource/{id1}</code> • <code>/messages/{messageid}/comments/{commentid}</code> and <code>/messages/{messageid}/likes/{likeid}</code> where the paths differ after <code>{messageid}</code>.
Method	<p>The REST operation which the flow implements, for example GET, PUT, POST, or DELETE. This is a non-editable field since each REST flow implements a single operation.</p>
Output Validation	<p>When set to True, the incoming data (body, headers, and query parameters) is validated against the configured JSON schema. By default, it is set to False.</p>

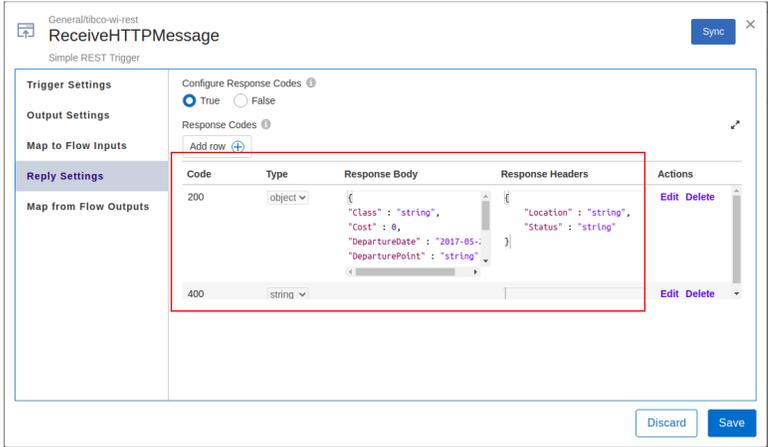
Output Settings

Field	Description
Query Parameters	<p>Query parameters to be appended to the path. To add the query parameters, click the  button and press Enter to save your changes.</p> <p>parameterName: Name of the query parameter</p> <p>type: The data type of the query parameter. Supported types are string, number, or boolean.</p> <p>repeating: Set to True if more than one value is expected for the query parameter.</p> <p>required: Set to True if query parameter is a required configuration. The trigger reports an error if no values are provided to the required query parameter.</p>
Path Parameters	Path parameters that are appended to the path.
Headers	<p>Header values for the trigger. To add the header parameters, click the  button and press Enter to save your changes.</p> <p>parameterName: Name of the header parameter.</p> <p>type: The data type of the header parameter. Supported types are string, number, or boolean.</p> <p>repeating: Set to True if more than one value is expected for the HTTP header.</p> <p>required: Set to True if header parameter is a required configuration. The trigger reports an error if no values are provided to the required header parameter.</p>
Request Schema	Request schema for the trigger. Be sure to use straight quotes for element names and values in the schema.

Map to Flow Inputs

This tab allows you to map the trigger output to flow input.

Reply Settings

Field	Description
Configure Response Codes	<p>Allows you to configure response codes.</p> <p>Default: False (See "Reply Data Schema" in this table.)</p> <p>To specify a response code, select True and click the  button. Enter the following details:</p> <ul style="list-style-type: none"> • Code: Enter the response code. • Type: Select the type of response expected for the Code. Supported types are String and Object. • Response Body: If Object is selected as the Type, enter the JSON schema in the Response Body column. For String, you need not enter anything in the Response Body column. • Response Headers: The header parameters for the reply in JSON data format. • Actions: The actions displayed change based on the type of the response code. <ul style="list-style-type: none"> – Edit, Delete: For an Object type of response, you can edit the details or cancel it. – Save, Cancel: For a String type of response, you can save or cancel the changes. <p>The response codes appear in the Map from Flow Outputs tab.</p> <p> The REST reply data type is by default set to data type any. To configure the reply to an explicit data type, see "Configuring the REST Reply" section in the <i>TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide</i>.</p> <p>For multiple response codes, use the ConfigureHTTPResponse activity in the flow to map Response Body and Response Headers from the REST trigger with the Input in the activity. To configure ConfigureHTTPResponse activity, see ConfigureHTTPResponse.</p> <p>The image shows the Reply Settings with multiple response codes.</p> 

Field	Description
	 <p>If you modify the response code schema in the table, the corresponding ConfigureHTTPResponse activities within that flow do not change appropriately. This happens specifically when removing fields from the Reply Settings tab. Redo the mappings for the ConfigureHTTPResponse activities.</p>
Reply Data Schema NOTE: This field appears only when Configure Response Codes is set to False .	The schema used for the reply data of the trigger. Be sure to use straight quotes for element names and values in the schema.

Map from Flow Outputs

Map the flow output to the trigger reply in this tab.



To update these settings for a trigger configured from Swagger 2.0 or OpenAPI 3.0 specification, make changes in the API specification file and upload it in the **Trigger Settings**. Do not update the settings as manual updates are removed.

GraphQL Trigger

The **GraphQL Trigger** lets the Flogo app act as the GraphQL server. To use this trigger, you simply upload your GraphQL schema and TIBCO Cloud Integration - Flogo (PAYG) automatically creates the flows corresponding to each query or mutation field in your schema.

Trigger Settings

Field	Description
Trigger Settings	
Port	The port on which the trigger listens to requests. By default, it is set to 7879. You can change this to use any other port that is open. This field can also be set using an application property.  <p>If the app has multiple triggers that require a port to be specified, make sure that the port number is unique for for each trigger. Two triggers in the same app cannot run on the same port.</p>
Path	The HTTP resource path for the operation. By default, it is set to <code>/graphql</code> , but you can change it to any string that is meaningful to you. It is the single endpoint that GraphQL queries and mutations use to access data from the multiple resources on the server. This field can also be set using an application property.

Field	Description
GraphQL Schema File	<p>The file that contains the GraphQL schema used to create the flow. The file has a .gql or .graphql extension.</p> <p> If you have made changes to the GraphQL schema file that you uploaded when creating the flow or trigger, you must propagate the changes to the flow input and flow output. To do this, after you select the updated schema file in this field, click the Sync button on the top right corner.</p>
Secure Connection	<p>By default, it is set to False. If you set this field to True, you can create a secure endpoint by providing Server Key and CA or Server Certificate.</p> <p>Server Key - A PEM encoded private key file</p> <p>CA or Server Certificate - A PEM encoded CA or self-signed server certificate file</p>
Handler Settings	
GraphQL Operation	The type of Graphql operation the flow should represent. You can select either Query or Mutation
Resolver for	This field is populated based on the type of GraphQL Operation that you selected. If you selected Query , the Resolver For lists the field names under the query type in the schema. If you select Mutation , the drop down menu lists the field names under the mutation type in the schema.

Map to Flow Inputs

This tab allows you to map the trigger output to flow input. The tab contains an element, `arguments`, which contains a list of fields or objects that match the input arguments of the **Resolver For** field in the GraphQL schema.

Map from Flow Outputs

Map the flow output to the trigger reply in this tab. The tab contains a child element, `data`, which contains either a simple type or an object that match the output type of the **Resolver For** field in the GraphQL schema. If the output type of the field is an `interface` type, the data will contain a single field of type `any`.

gRPC Trigger

The gRPC trigger acts as the server to the gRPC clients. You create the trigger when you create a flow that gets attached to the trigger.

Trigger Settings

Field	Description
Trigger Settings	
Port	The port on which the trigger listens to requests. You can use any port that is open. This field can also be set using an application property.

Field	Description
Proto File	<p>A file with <code>.proto</code> extension that contains the methods and service(s) definition which TIBCO Cloud Integration - Flogo (PAYG) uses to create the flows. Currently, importing a <code>.proto</code> file into another <code>.proto</code> file is not supported.</p>  <ul style="list-style-type: none"> The gRPC trigger and gRPC activity does not support options in the <code>.proto</code> file. If your <code>.proto</code> file contains any options, be sure to remove the options in <code>.proto</code> file before using it. You must not use the same gRPC <code>.proto</code> file for a gRPC trigger and gRPC activities in the same app. The package names for the gRPC trigger and gRPC activities must be unique.
Secure Connection	<p>By default, it is set to False. If you set this field to True, you can create a secure endpoint by providing Server Key and CA or Server Certificate.</p> <p>Server Key - A PEM encoded private key file</p> <p>CA or Server Certificate - A PEM encoded CA or self-signed server certificate file</p>
Handler Settings	
Service Name	Name of the service defined in the <code>.proto</code> file. You must create one gRPC trigger for any specific <code>.proto</code> file. Any subsequent gRPC triggers using the same <code>.proto</code> file can select the service and method they need from the dropdowns.
Method	Name of RPC method in the <code>.proto</code> file. Each method in the <code>.proto</code> file is represented by a separate flow and attached to the same gRPC trigger.

Map to Flow Inputs

This tab allows you to map the trigger output to flow input. The tab displays fields from your selected method.

Map from Flow Outputs

Map the flow output to the trigger reply in this tab.

Activities

In addition to the activities available for general use, TIBCO Cloud Integration - Flogo (PAYG) supports activities that were originally created in Project Flogo™. Such activities are marked with an OSS tag on them. This allows for a seamless import of apps that were created in Project Flogo™. The Project Flogo™ activities are placed under the **Default** category.

If you are creating an app in TIBCO Cloud Integration - Flogo (PAYG), it is preferable to use the general purpose activities (the activities that do not have an OSS tag on them), since they are more robust in functionality.

Refer to <https://github.com/TIBCOSoftware/flogo-contrib> for details on the activities that are marked with an OSS tag.

The available activities are placed under the following categories:

- Default
- General

You can create a flow branch from any activity except the **Return** activity.

This section contains the following topics:

ConfigureHTTPResponse

This activity is used to configure HTTP response codes that you want to use in your REST reply.

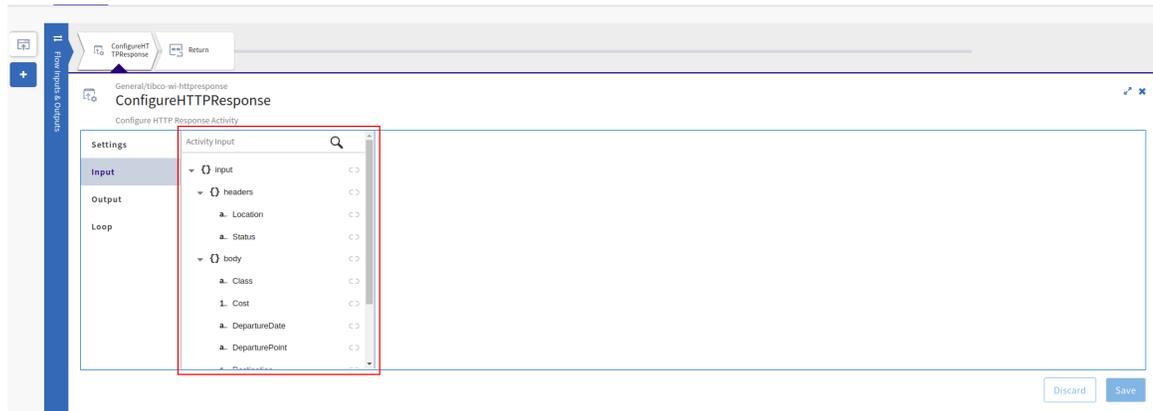
Settings

Field	Description
Trigger Name	The list of triggers to which this flow is attached. This field is displayed only when the flow it attached to multiple REST triggers. Select a trigger in which the code you want to use is configured.
Code	List of response codes that you have configured in the selected trigger. Select a response code you want to use.

Input

Displays the schema for the code that you selected in the Settings tab. Map the elements in the schema in the mapper or manually enter the values that you want to send as a response.

For multiple response codes in the [ReceiveHTTPMessage](#) REST trigger, map the **Response Body** and **Response Headers** from the trigger with the **Input** in this activity. The image shows the **Input** with **headers** and **body** mapped from the REST trigger response codes.



Output

Displays the schema for the code in a read-only tree format.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

gRPCActivity

The gRPC Invoke activity is an implementation of the gRPC client. Use this activity to make outbound gRPC calls.

Settings

Field	Description
Host URL	URL used to connect to the gRPC server. The name or IP address of the host on which your .proto file resides. Be sure to append the port number on which the service is running.
Secure Connection	By default, it is set to False . If you set this field to True , you can create a secure endpoint by providing Server Key and CA or Server Certificate . Server Key - A PEM encoded private key file CA or Server Certificate - A PEM encoded CA or self-signed server certificate file
Proto File	A file with .proto extension that contains the methods and service(s) definition which TIBCO Cloud Integration - Flogo (PAYG) uses to create the flows. Each flow corresponds to one method in the .proto file. Currently, importing a .proto file into another .proto file is not supported.  <ul style="list-style-type: none"> The gRPC trigger and gRPC activity does not support options in the .proto file. If your .proto file contains any options, be sure to remove the options in .proto file before using it. You must not use the same gRPC .proto file for a gRPC trigger and gRPC activities in the same app. The package names for the gRPC trigger and gRPC activities must be unique.
Service Name	Name of the service you want to invoke. The service is defined in the .proto file that you have selected.
Method	Name of RPC method in the selected .proto file. Each method in the .proto file is a gRPC request which is represented by a separate flow and attached to the same gRPC trigger.

Input

Field	Description
Activity Input	The Input tab lists the parameters for the method that you chose in the Settings tab so that you can either enter or map their values in the mapper.

Output

The **Output** tab displays a read-only schema of the activity output (the response that is configured for the selected method).

Iterator

For details on using the iterator, see the "Using the Iterator in an Activity" section in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide*.

InvokeLambdaFunction

Use this activity to invoke a specific Lambda function.

Settings

Field	Description
AWS Connection Name	Select a AWS connection
ARN (Optional)	<p>Amazon Resource Name.</p>  <ul style="list-style-type: none"> You can also specify the ARN in the Input tab. If you specify the ARN in both the tabs, the ARN in the Input tab is used. You must specify the ARN in at least one tab; otherwise the activity returns an error at runtime.

Input Settings

Field	Description
Payload Schema	Enter a JSON request schema for your payload that will be used to invoke the Lambda function.

Input

The payload schema that you entered in the **Input Settings** tab is displayed in a tree format in the **Input** tab. Map the elements in the schema using the mapper or alternatively, enter values for the element by manually typing the value in the mapper.

Field	Description
LambdaARN	<p>Amazon Resource Name.</p>  <ul style="list-style-type: none"> You can also specify the ARN in the Settings tab. If you specify the ARN in both the tabs, the ARN in the Input tab is used. You must specify the ARN in at least one tab; otherwise the activity returns an error at runtime.

Output Settings

Field	Description
Result Schema	The schema for the result that is expected from the Lambda function invoke request

Output

The Output tab displays the result schema you entered in the **Output Settings** tab in a tree format.

InvokeRESTService

This activity is used to make a request to a REST service; it also accepts the reply returned by the service.

Settings

Field	Description
API Spec	<p>(Optional) Click Browse and browse to the file location on the machine. Select a JSON file.</p> <p>Supported specifications are Swagger Specification 2.0 and OpenAPI Specification 3.0.</p>
Resource Path	<p> This field is displayed only if you upload a JSON file in the API Spec field.</p> <p>All resource paths available in the JSON file are listed in the dropdown. Depending on the resource path you choose, the supported operations are listed in the Method field.</p>
Method	Select an operation for the request. For example: GET, POST, PUT, DELETE, or PATCH.
URL	<p>An absolute path to the REST service that you want to invoke. For example: <code>http://acme.com</code> or <code>https://acme.com</code>. If you enter an absolute path beginning with <code>https://</code>, the Use certificate for verification section appears.</p> <p>If your connection requires an SSL certificate, select True. Otherwise, select False. To add a certificate, under Server Certificate, click Browse and browse to the certificate location on the machine.</p> <p> If you upload an OpenAPI 3.0 JSON specification file in the API Spec field, the URL is a drop down menu. This lists the server URLs mentioned in the JSON file. Select a server URL from the list.</p>
Timeout	<p>Specify the timeout period (in milliseconds) for invoking a service. If a timeout value is specified, the activity waits for the specified time. If a response is not received by the specified time, the request expires with an error.</p> <p>Default: 0 milliseconds (that is, there is no timeout for invoking a service)</p>
Request Type	<p> This field is displayed only for the POST, PUT, and PATCH methods.</p> <p>The Request content type of the REST service. The following content-type are supported:</p> <ul style="list-style-type: none"> • <code>text/plain</code> • <code>application/json</code> • <code>application/x-www-form-urlencoded</code> <p> If you select <code>application/x-www-form-urlencoded</code>, the default schema is set in the Request Schema field of the Input Settings tab. You can edit the default schema or specify your own schema. If you specify your own schema, it must be a name-value string pair.</p>

Field	Description
Proxy URL	<p>Specify the URL to the HTTP proxy server. If a proxy URL is specified, the request to the REST service (specified in the URL field) is routed via this proxy URL.</p> <p> A secure connection to the proxy server is not supported.</p> <p>Default: Proxy URL is disabled.</p>

Input Settings



If you upload a JSON file in the **API Spec** field, the fields in **Input Settings** are automatically populated according to the **Resource Path** you select.

Field	Description
Query Params	<p>Query parameters to be appended to the path. To add the query parameters, click the  button and press Enter to save your changes.</p> <p>parameterName: Name of the query parameter.</p> <p>type: The data type of the query parameter. Supported types are string, number, or boolean.</p> <p>required: Set to True if query parameter is a required configuration. The trigger reports an error if no values are provided to the required query parameter.</p>
Path Params	<p>Path parameters that are appended to the path. This is a non-editable field.</p> <p>parameterName: Name of the path parameter. This is the parameter specified in between { } in the Resource Path field or the URL field in Settings.</p> <p>type: The data type of the path parameter. Supported type is string.</p>
Request Headers	<p>Header values for the InvokeRESTService activity. To add the header parameters, click the  button and press Enter to save your changes.</p> <p>parameterName: Name of the header parameter.</p> <p>type: The data type of the header parameter. Supported types are string, number, or boolean.</p> <p>required: Set to True if header parameter is a required configuration. The trigger reports an error if no values are provided to the required header parameter.</p>
Request Schema	<p>Enter a request schema here. This field is visible only if you selected the POST, PUT, or PATCH method in the Settings tab.</p> <p> If you selected <code>application/x-www-form-urlencoded</code> as the Request Type in the Settings tab, the default schema is set here. You can edit the default schema or specify your own schema. If you specify your own schema, it must be a name-value string pair.</p>

Input



If you upload a JSON file in the **API Spec** field, the **Input** fields are automatically populated according to the **Resource Path** and **Method** you select.

Field	Description
host	Specify the value which will override <code>hostname:port</code> value specified in the URL at runtime with the value specified in this configuration. Enter a value in the form <code>hostname[:port]</code> where <code>[:port]</code> is optional.
queryParams	Provide a value to the query parameters configured on the Input Settings section. This field is visible only if you selected the POST or PUT method in the Settings tab.
pathParams	Provide a value to path parameters defined as part of URL in the Settings tab. This field is visible only if you selected the POST or PUT method in the Settings tab.
headers	Header values for the activity. These values can be manually entered or mapped to the output of the trigger or any preceding activity.
body	Request Schema values for the activity. These values can be manually entered or mapped to the output of the trigger or any preceding activity. This field is visible only if you selected the POST or PUT method in the Settings tab.

Output Settings



If you upload a JSON file in the **API Spec** field, the fields in **Output Settings** are automatically populated according to the **Resource Path** you select.

Field	Description
Configure Response Codes	<p>Allows you to configure response codes.</p> <p>Default: False (See "Response Schema" and "Response Type" in this table.)</p> <p>To specify a response code, select True and click the  button. Enter the following details:</p> <ul style="list-style-type: none"> • Code: Enter a specific response code or configure a single schema for a category of response codes. For example, if all the status codes are similar (such as 501, 502, 503, and so on), you can define a single schema (as 5xx) for them. Defining a single schema saves you time and effort as you do not need to configure each status code separately in the activity. <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-left: 20px;"> <p> If the status code is provided as a range (5xx in the above example) and also an absolute format (501 in the above example), the status code in the absolute format is given priority. In the above example, status code 501 is given priority over 5xx at runtime.</p> </div> <ul style="list-style-type: none"> • Type: Select the type of response expected for the Code. Supported types are String and Object. • Response Body: If Object is selected as the Type, enter the JSON schema in the Response Body column. For String, you need not enter anything in the Response Body column. • Actions: The actions displayed change based on the type of the response code. <ul style="list-style-type: none"> – Edit, Delete: For an Object type of response, you can edit the details or cancel it. – Save, Cancel: For a String type of response, you can save or cancel the changes. <p>The response codes appear in the Output tab.</p>
Response Schema	<p> This field appears only when Configure Response Codes is set to False.</p> <p>The schema for the reply that the server sends.</p>
Response Type	<p> This field appears only when Configure Response Codes is set to False.</p> <p>The content type of the REST service. The following content-types are supported:</p> <ul style="list-style-type: none"> • text/plain • application/json • other
Response Headers	<p>The header parameters for the reply.</p>

Output

The Output tab displays the headers and response body configured for both the request and the response in a tree format.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide for information on the **Loop** tab.

Retry on Error

This tab allows you to set the number of times the flow should try to execute the activity if it encounters a retrievable error (such as waiting for a server to start, intermittent connection failures, or connection timeout) during the activity execution.

Map the elements in the schema using the mapper or alternatively, enter values for the element by manually typing the value in the mapper. See the section on "Using the Mapper" in *TIBCO Flogo® Enterprise User's Guide* for details on how to map elements.

Field	Description
Count	The number of times the flow should attempt to execute the activity.
Interval	The time (in milliseconds) to wait in between each attempt to execute the activity.



To update these settings for an activity configured from Swagger 2.0 or OpenAPI 3.0 specification, make changes in the API specification file and upload it in **Settings**. Do not update the activity settings as manual updates are removed.

Using SSL

If you choose to set up SSL authentication for the **InvokeRESTService** activity, you must have a self-signed server certificate that you must upload when setting up the activity.



Use Self-signed PEM certificate for secure connection.

To set up SSL authentication, follow these steps:

Prerequisites

You must have the self-signed server certificate handy on your machine.

Procedure

1. On the flow page, click the **Invoke REST Service** tile to open its properties.
2. In the **Settings** tab, under **Use certificate for verification**, select the **True** radio button. This exposes the **Browse** button. The SSL verification will be turned off when **Use certificate for verification** is set to **False**.
3. Use the **Browse** button to navigate to the location of the server certificate. Once the server certificate is uploaded successfully, the connection uses the certificate to authenticate.

LogMessage

LogMessage is an activity that writes a message to the log. For each application, there is a log file. You can view the logs in the **Log** tab.

You can view the logs in the **Logs** tab.

Settings

The **Settings** tab has the following fields.

Field	Description
Log Level	Select one of the following log levels: <ul style="list-style-type: none"> • Info: logs informational messages highlighting the application progress. • Warning: is the warning message of an unexpected error when running the flow. • Error: logs error conditions and messages. • Debug: can be used for debug-level messages.
Add Flow Details	Appends Flow Instance ID, Flow Name and Activity Name to the log message. By default, this field is set to False .

Input

Provide the following input for this activity.

Input Item	Description
message	The message to be displayed in the log.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Mapper

Use this activity to define a schema to get the desired data. This activity is particularly useful to define a schema for an object of type any. In the flow, you place the Mapper activity preceding an activity whose input requires an object of data type any. This allows you to map the object of type any to the output from the Mapper activity. An advantage of using this activity is that you can construct the data for the any data type within the flow instead of fetching it from outside.

Input Settings

Field	Description
Input Schema	Enter the JSON schema that will be used as the input for this activity. The elements of this schema are available for mapping in the Input tab and are mappable to the output from any preceding activity, trigger, or the flow input. The Mapper activity outputs the elements from this schema, so they are also displayed in the Output tab in a tree format. This makes them available for mapping in the following activities.

Input

The **Input** tab displays the schema you entered in the **Input Settings** tab in a tree format. You can map these elements to the output from any preceding activity, trigger, or the flow input.

Output

The **Output** tab displays the elements from the schema you entered in the **Input Settings** tab.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

ParseJSONActivity

This activity takes a stringified JSON data as input and converts it into a JSON object, which can then be accessed by the downstream activities that follow. You provide the input to the activity either by entering the stringified JSON data manually in the **Input** tab or saving it in a file and entering the file path in the **Input** tab. The activity supports output validation if you opt to validate the input JSON data against the output schema that you configure in the **Output Settings** tab.

Settings

Field	Description
Output Validation	<p>True: Select True to validate that the JSON data in your input string matches the schema that you configure in the Output Settings tab of the activity.</p> <p>False: Select False if you do not want to validate the JSON data in your input string against the schema you configured.</p> <p>This field can be configured with an application property.</p>

Input

Field	Description
jsonString	The string containing the JSON data that you want to parse. This activity creates a JSON object with the parsed JSON data. Enter the string manually or map it to an element from the output of the trigger, flow input, or one of the preceding activities.

Output Settings

Field	Description
Schema	The schema that you want to use to create the JSON object. You have the option to validate the stringified JSON (input to the activity) against this schema.

Output

The output schema is displayed in a read-only tree format.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

ReplyToHTTPMessage (Supported for backward compatibility only)

This activity is not supported in TIBCO Cloud Integration - Flogo (PAYG) 2.5.0 and above. This activity is applicable only to flows that were created in previous versions of TIBCO Cloud Integration - Flogo (PAYG) (prior to version 2.5.0) that are imported into the current version.

This activity automatically gets created when you create a flow with a REST trigger. It is used by the server to reply to a request from the REST client.

Configuration

Field	Description
Reply	<p>Reply sent by the server in response to the REST client request. The two supported replies are Success with Data and Error with Message.</p> <p>If you select Success with Data, the reply schema must be configured in the Input Settings tab in the Schema field. If you select Error with Message, you must configure the error message in the message field of the Input tab.</p>

Input Settings

Field	Description
Schema	Enter the reply schema or sample data using a JSON structure.

Input

Field	Description
message	The string that is included in the reply. If you configured the Reply field in the Configuration tab with Error with Message , the error message must be entered in message text box. If you configured the Reply field with Success with Data , then you must map your data according to the schema specified in the Input Settings tab.

SendMail

SendMail is an activity that sends an email by way of an SMTP server.



To securely configure the **SendMail** activity using the `smtp.gmail.com` server, use TLS on Port 587 or SSL on port 465.

Settings

The **Settings** tab has the following fields.

Field	Description
Server	The host name or IP address for the mail server.
Port	The port used to connect to the server.

Field	Description
Connection Security	The type of connection to be used to communicate with the mail server. Select TLS or SSL depending upon the security configuration of the mail server. In case no security is enabled on the mail server, select NONE.
Username	The username to use when authenticating to the mail server.
Password	The password to use when authenticating to the mail server.

Input

This tab displays the fields that are used as input for the activity.

Input Item	Description
message_content_type	The type of message content. Valid types are "text/plain" or "text/html".
sender	The email address of the sender.
recipients	The recipient list for the email. You can send the mail to multiple recipients. Provide a list of recipients in a single string by using a comma as the delimiter.
cc_recipients	The CC recipient list for the email. You can send the mail to multiple recipients. Provide a list of recipients in a single string by using a comma as the delimiter.
bcc_recipients	The BCC recipient list for the email. You can send the mail to multiple recipients. Provide a list of recipients in a single string by using a comma as the delimiter.
reply_to	Email address to which the reply message is to be sent.
subject	The subject of the email.
message	The content of the email message.

Input Item	Description
attachments	<p>File attachments to be sent along with the email message.</p> <p>To map the child elements, add <code>array.forEach()</code> to the <code>attachments</code> field and then specify the child elements as follows:</p> <ul style="list-style-type: none"> <code>file</code>: Specify the path of the file to be attached using <code>file://<path></code> or specify the content of the file by enclosing it in double quotes. <ul style="list-style-type: none">  In Flow Tester, <code>file://<path></code> cannot be specified. <code>filename</code>: Specify the name of the file to be attached. <code>base64EncodedContents</code>: If the file is a Base64 encoded file, set this field to <code>true</code>. The default is blank (or <code>false</code>). <p>To send multiple attachments, use the Loop feature.</p>

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Sleep

The Sleep activity is an asynchronous activity that suspends the execution of a flow for the specified time.

Settings

Field	Description
Interval Type	<p>The unit of the time interval for which the execution of the flow must be suspended. Supported types are Millisecond, Second, and Minute.</p> <p>Default: Millisecond</p>
Interval	<p>The time interval for which the execution of the flow must be suspended.</p> <p>Default: 0</p>

Input



The fields in the **Input** tab are required only if you need to pass values from the output of a previous activity or trigger. Otherwise, you can directly specify the values in the **Settings** tab. Values specified in the **Input** tab take precedence over values specified in the **Settings** tab, if values are configured in both the tabs.

Field	Description
Interval Type	<p>The unit of the time interval for which the execution of the flow must be suspended. Supported types are Millisecond, Second, and Minute.</p> <p>Default: Millisecond</p>

Field	Description
Interval	The time interval for which the execution of the flow must be suspended. Default: 0

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Apache Avro

This section contains the following topics:

Overview

TIBCO Cloud Integration - Flogo (PAYG) provides out-of-the-box activities to serialize and deserialize Avro messages. It uses JSON-based schemas. For information on using Apache Avro refer to the Apache Avro documentation.

Avro Serialize

Use this activity to serialize the message data into a base64 encoded string using the schema and message that you configure in the **Input** tab of this activity. Once the message is base64 encoded, it can be transported over network.

Settings

The elements from the schema that you enter in this tab will be available for manually configuring or mapping in the **Input** tab.

Field	Description
Schema	The Apache Avro schema to be used to serialize a message. It contains the complex and/or primitive types that are supported by Apache Avro.

Input

The **Input** tab displays the elements of the schema that you entered in the **Settings** tab in a tree format. You can input the values for each element by hard coding the value or mapping the value to an element from the output schema of a previous activity in the flow.

Output

This tab displays serialized data in the base64 encoded format.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Avro Deserialize

Use this activity to deserialize serialized data into a JSON readable format using the schema that you configure in the **Settings** tab of this activity.

Settings

The elements from the schema that you enter in this tab will be available for manually configuring or mapping in the **Input** tab.

Field	Description
Schema	The Apache Avro schema to be used to deserialize a message. It contains the complex and/or primitive types that are supported by Apache Avro.

Input

Field	Description
serializeddata	Enter the serialized data or map it to serialized data from the output of a previous activity. This data will be decoded into JSON format.

Output

The **Output** tab displays the elements of the schema that you entered in the **Settings** tab in a tree format after the serialized data from the **Input** tab has been deserialized.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Apache Kafka

This section contains the following topics:

Overview

Apache Kafka is a distributed messaging system, providing fast, highly scalable, and redundant messaging through a publisher-subscriber model. By using out-of-the-box connectivity, you can design the flows to send and receive the records.



For information about how to use Apache Kafka, see [Kafka documentation](#).

Configuring a Kafka Client Connection

To publish or subscribe messages, you must first configure a Apache Kafka client connection. The Apache Kafka client connection contains the parameters required to connect to the Apache Kafka cluster. The Apache Kafka client connection is used by all the activities in the Apache Kafka category.

Prerequisites

Before you create a client connection, familiarize yourself with Apache Kafka. For details about how to use the Apache Kafka product, see the [Kafka Documentation](#).

Procedure

1. On the TIBCO Cloud™ Integration - Flogo® (PAYG) page, click the **Connections** tab and perform one of the following actions:
 - To add a client connection for the first time, click the **Apache Kafka Client Configuration** card. You can search for a connector card by typing the connector name in the search field.
 - If you have existing connections and want to add a new connection, click the **Add Connection** link.
2. In the Apache Kafka Client Configuration dialog box, enter the connection details. For field descriptions, see the [Kafka Connection Details](#) topic.
3. Click **Save**.

Kafka Client Configuration Details

To establish the connection successfully, you must configure the Apache Kafka instance.

The **Apache Kafka Client Configuration** dialog box contains the following fields:

Condition Applicable	Field	Description
N/A	Connection Name	The unique name for the connection you are creating. This name is displayed in the Connection drop-down list for all the TIBCO Flogo® Connector for Apache Kafka activities.
N/A	Description	A short description of the connection
N/A	Brokers	A comma-separated list of host and port pair (host:port) for establishing the initial connection with the Kafka cluster.
Applicable only when SASL/PLAIN is selected in the Auth Mode field.	Auth Mode	Select one for following authentication type to establish the connection with Kafka cluster: <ul style="list-style-type: none"> • None: To establish the connection without authentication • SASL/PLAIN: To use Simple Authentication Security Layer (SASL) PLAIN authentication • SSL: To use Secure Socket Layer (SSL) authentication
	User Name	The user name for authentication.
	Password	The password for authentication.
Applicable only when SASL/PLAIN or SSL is selected in the Auth Mode field.	Client Certificate	A Privacy Enhanced Mail (PEM) encoded client certificate file for mutual authentication.
	Client Key	A PEM encoded private key file for mutual authentication.
	CA or Server Certificate	A PEM encoded private key file for server authentication.

Condition Applicable	Field	Description
	Connection Timeout	The amount of time in seconds to wait for the initial connection. Default value: 30 seconds
N/A	Retry Backoff	The amount of time in milliseconds to wait for leader election to occur before retrying. Default value: 250 milliseconds
N/A	Max Retry	The number of attempts to retry metadata request when the cluster is in the middle of a leader election. Default value: 3 attempts
N/A	Refresh Frequency	The amount of time in seconds after which metadata is refreshed. Default value: 40 seconds
N/A	Use Schema Registry	<p>Enables you to use the Avro schema with a schema registry by selecting True.</p> <p>If you select True, you must specify a Schema Registry URL. You can use the Confluent schema registry or the TIBCO schema registry. Depending on which schema registry you want to use, the Schema Registry URL will need to change. Basic authentication is used to connect to the registry if a Username and Password are provided.</p> <ul style="list-style-type: none"> • If True is selected, the schema displayed in the Schema for Avro Value field in the following tabs is read-only. If False is selected, you can create or modify the schema in both these tabs. <ul style="list-style-type: none"> – Input Settings tab of Kafka Producer – Output Settings tab of Kafka Consumer Trigger • SSL authentication is not supported. <p>Default value: False</p>
Applicable only when True is selected in the Use Schema Registry field.	Schema Registry URL	<p>The URL used to connect to a schema registry.</p> <p>For the Confluent schema registry, the URL must be in the following format:</p> <pre>http://<host>:<port></pre> <p>For the TIBCO schema registry, the URL must be in the following format:</p> <pre>http://<host>:<port>/schema/v1</pre> <p> For Tibco Schema Registry use the FTL realm URL.</p>

Condition Applicable	Field	Description
	Username	(Optional) The username to access the schema registry with basic authentication.
	Password	(Optional) The password to access the schema registry with basic authentication.

Kafka Consumer Trigger

Apache Kafka Consumer Trigger receives records from specified topic in the Apache Kafka cluster.

Trigger Settings

On the **Settings** tab, you can define the Apache Kafka connection and its details as given in the following table:

Condition Applicable	Field	Description
N/A	Apache Kafka Client Configuration	Apache Kafka client configuration to be used.
N/A	Topic	The topic where Apache Kafka cluster stores streams of record.  If you are using multiple topics, separate them using commas.
N/A	Topic Whitelist	(Optional) A whitelist of topics you can subscribe to for consuming messages from multiple topics. You can specify the group using a regex pattern. For example: <code>mytopic*</code>
N/A	Consumer Group ID	The group ID for the consumer group.
N/A	Value Deserializer	Select the type of record value to be received from the drop-down list: <ul style="list-style-type: none"> • String • JSON • Avro
Applicable only when Avro is selected in the Value Serializer field.	Subject	A list of all registered subjects in your schema registry. A subject refers to the name under which a schema is registered. Select the subject to be used.

Condition Applicable	Field	Description
Applicable only when Avro is selected in the Value Serializer field.	Version	Version of the subject (registered name of schema) registered. Select the version of the subject to be used.
N/A	Commit Interval	The time interval in which a consumer offset commits to Apache Kafka. Default Value: 5000 milliseconds
N/A	Initial Offset	Select one of the following options: <ul style="list-style-type: none"> • Newest: To start receiving published records since the consumer is started • Oldest: To start receiving records since the last commit
N/A	Fetch Min Bytes	Minimum size of data that server sends on fetch request.
N/A	Fetch Max Wait	The maximum amount of time that the server would block before answering a fetch request if there is not sufficient data to immediately satisfy the requirement that you have configured in the Fetch Min Bytes field.
N/A	Heartbeat Interval	Time in milliseconds to send heartbeats to consumer. Heartbeats are used to ensure that the consumer's session remains active and to facilitate rebalancing when consumers join or leave a group.  Heartbeat interval must not be more than one-third of the session time.
N/A	Session Timeout	The consumer sends periodic heartbeats to server indicating about its liveness to the broker. If no heartbeats are received by a broker before the session times out, the broker removes this consumer from the group and initiates a rebalance.

Output Settings

Condition Applicable	Field	Description
N/A	Headers	Record headers to be received. Only <code>String</code> datatype value is supported.  Headers are supported in the Apache Kafka version 0.11.0 and later.
Applicable only when JSON is selected in the Value Serializer field on the Trigger Settings tab.	Schema for JSON value	The JSON schema for the Apache Kafka record value

Condition Applicable	Field	Description
Applicable only when Avro is selected in the Value Serializer field on the Trigger Settings tab.	Schema for Avro Value	<p>The Avro schema for the Apache Kafka record value. Depending on the Subject and Version selected, the schema is displayed here.</p> <p> This field is read-only if Use Schema Registry in the Apache Kafka Client Configuration dialog box is set to True. Otherwise, you can provide the schema using this editor.</p>

Output

Condition Applicable	Field	Description
Applicable only when JSON is selected in the Value Serializer field.	jsonValue	Data structure based on JSON schema that you have configured in the Output Settings section.
Applicable only when Avro is selected in the Value Serializer field.	avroData	Data structure based on Avro schema that you have configured in the Output Settings section.
N/A	partition	Partition number of the record
N/A	offset	Offset of the record
N/A	topic	Name of the topic
N/A	key	Key value
Applicable only when String is selected in the Value Deserializer field on the Settings tab.	stringValue	String value to be received
N/A	headers	Header value to be received

Kafka Producer

Apache Kafka producer activity sends a record to a specified topic or channel in the Kafka cluster.

Settings

On the **Settings** tab, you can define the Apache Kafka connection and its details as given in the following table:

Condition Applicable	Field	Description
N/A	Apache Kafka Connection	Select the connection you want to use from the drop-down list.

Condition Applicable	Field	Description
N/A	Acks Mode	Select one of the following acknowledgement modes from the drop-down list: <ul style="list-style-type: none"> • None: To receive no acknowledgement on record delivery • Leader: To receive an acknowledgement on record delivery from the leader • All: To receive acknowledgement on record delivery from leaders and all in-sync replicas
Applicable only when All is selected in the Ack Mode field.	Ack Timeout	The amount of waiting time in milliseconds to receive confirmation.
N/A	Compression Type	Select a compression type: None , GZIP , or LZ4 .
N/A	Value Serializer	Select the type of record value to be sent: <ul style="list-style-type: none"> • String • JSON • Avro
Applicable only when Avro is selected in the Value Serializer field.	Subject	A list of all registered subjects in your schema registry. A subject refers to the name under which a schema is registered. Select the subject to be used.
Applicable only when Avro is selected in the Value Serializer field.	Version	Version of the subject (registered name of schema) registered. Select the version of the subject to be used.
N/A	Max Request Size	The maximum size of buffered records that can be sent in one request. Default value: 1048576 bytes
N/A	Max Messages	The maximum number of records that can be sent in a single broker request.
N/A	Frequency	The frequency of sending buffered records in milliseconds. Default value: 1000

Input Settings

Condition Applicable	Field	Description
N/A	Headers	Header record to be sent. Only String datatype value is supported.  Headers are supported in the Apache Kafka version 0.11.0 and later.
Applicable only when JSON is selected in the Value Serializer field on the Settings tab.	Schema for JSON value	The JSON schema for the Apache Kafka record value.
Applicable only when Avro is selected in the Value Serializer field on the Settings tab.	Schema for Avro Value	The Avro schema for the Apache Kafka record value. Depending on the Subject and Version selected, the schema is displayed here.  This field is read-only if Use Schema Registry in the Apache Kafka Client Configuration dialog box is set to True . Otherwise, you can provide the schema using this editor.

Input

Condition Applicable	Field	Description
N/A	topic	Name of the topic.
N/A	partition	Partition number of the record to be sent.
N/A	key	Optional key value.
Applicable only when String is selected in the Value Serializer field.	stringValue	String value to be sent.
Applicable only when JSON is selected in the Value Serializer field.	jsonValue	Data structure based on JSON schema that you have configured on the Input Settings tab.
Applicable only when Avro is selected in the Value Serializer field.	avroData	Data structure based on Avro schema that you have configured on the Input Settings tab.
N/A	headers	Header value to be sent.

Output

Condition Applicable	Field	Description
N/A	topic	Name of the topic.
N/A	partition	Partition number of the record to send.
N/A	offset	Offset of the record.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Kafka Offset Commit

Apache Kafka Offset Commit activity notifies Kafka Consumer Trigger to commit given offset. This is useful in case you want offsets to be committed as soon as the record is processed in the flow. By default, offsets are committed only when flow is successfully executed.



This activity can be used only in conjunction with Kafka Consumer Trigger.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Apache Pulsar

This section contains the following topics:

Overview

TIBCO Flogo® Connector for Apache Pulsar provides out-of-the-box activities and triggers to send and receive messages. You can connect to your Apache Pulsar broker using transport layer security (TLS) or JSON Web Token (JWT).

Configure triggers that subscribe to messages published to a topic, including Exclusive, Shared, Failover, or KeyShared subscription types. Use activities to send messages to a topic by mapping string or JSON content, with optional compression.

Apache Pulsar is popularly used for the following purposes:

- To subscribe to messages produced by edge computing devices and perform transformation or analysis on them.
- To publish messages on any broker and topic in your network based on the state of the flow. You can send the results of the flow to a topic.



You should not use Linux/x86 option to build the application in TIBCO Flogo® Enterprise as Apache Pulsar works only on a 64 bit machine. You must install the GNU compiler collection (GCC) compiler on the Windows machine to generate windows executable pulsar application from builder-windows-amd64.exe tool on Flogo® Enterprise. For information about how to use Apache Pulsar, see Apache Pulsar documentation.

Creating an Apache Pulsar Connection

To use Apache Pulsar, you must configure an Apache Pulsar client connection. The Apache Pulsar connection is used by all the activities in the Apache Pulsar category. The Apache Pulsar connection contains the parameters required to connect to Apache Pulsar.

Prerequisites

Before you create a connection, familiarize yourself with Apache Pulsar. For details about how to use Apache Pulsar, see the Apache Pulsar documentation.

Procedure

1. On the global navigation bar, click the **Connections** tab and perform one of the following actions:
 - To add a connection for the first time, click the Apache Pulsar card. You can search for a connector card by typing the connector name in the search field.
 - If you have existing connections and want to add a new connection, click the **Add Connection** link.
2. In the Apache Pulsar dialog box, enter the connection details. For field descriptions, see the [Apache Pulsar Connection Details](#) topic.
3. Click **Save Connection**.

Apache Pulsar Connection Details

The Apache Pulsar connection is a container for all the client connection parameters used by Apache Pulsar Publisher activity and Apache Pulsar Subscriber trigger. To establish the connection successfully, specify the following connection details in the Apache Pulsar dialog box:

Field	Required	Description
Name	Yes	Specify a unique name for the connection you are creating. This name is displayed in the Connection Name drop-down list for each activity.
Description	No	A brief description of the connection.
Broker URL	Yes	<p>The URL of the broker. It follows the format- <code>pulsar+ssl://flex-linux-gazelle:2665</code> where <code>pulsar</code> is the protocol and <code>+ssl</code> is an optional parameter that shows that the connection is secured by TLS. The content following the <code>://</code> is the host name and the port on which the broker is running.</p> <p>If the URL of the broker is SSL then the CaCert field is enabled. This field must be provided unless the Allow Insecure Connection option is enabled.</p> <p>If the authorization is enabled, then the connector would insist that you use SSL to protect the credentials when you are on the network.</p>

Field	Required	Description
Authorization Type	Yes	You can choose the authorization type from the following options: <ul style="list-style-type: none"> None TLS: The client must present a certificate in which the Common Name field matches the role for resources requested on the broker. JWT: The client must present a JSON Web Token composed on the connected broker for the specific role authorized by this connection.
Allow Insecure Connection	Yes	The Allow Insecure Connection field is available if <code>+ssl</code> is present in the Broker URL field so that the self-signed development mode certificates can be used to connect to the broker. If this field is enabled, then the Broker CA field is removed as it is not needed anymore.
Broker CA	Yes	The Broker CA field points to the certificate authority used to sign the broker's certificate for secure connections. The Flogo client now trusts the broker.
Client Cert	Yes	If the Authorization Type field is TLS, then the Client Cert field must point to the client's certificate. The broker must trust this certificate.
Client Key	Yes	If the Authorization Type is TLS, then the Client Key field must point to the client's key.
JSON Web Token	Yes	If the Authorization Type is JWT, then the JSON Web Token field must contain the text contents of the JWT obtained for this connection.

Apache Pulsar Consumer Trigger

Apache Pulsar Consumer is a Flogo trigger that receives messages published to the configured topic. Each message triggers a new flow and if the flow is successful the message is acknowledged by removing it from the queue.

Trigger Settings

The **Trigger Settings** tab has the following fields:

Field	Required	Description
Pulsar Connection	Yes	Name of the connection.
Topic	Yes	The name of the topic from which the message is consumed. The Topic field has the following format <code>:persistent://public/default/foo</code> The first segment is the type of topic. The topic can be persistent or non-persistent. With persistent topics, all the messages are persisted to the disk. The second segment is the name of the domain or a tenant. The third segment is the namespace within the domain and the fourth segment is the name of the individual topic.

Field	Required	Description
Subscription Name	Yes	The subscription name is used by the broker to combine subscribers belonging to a single application into a logical group for message delivery. Subscriptions can also be used to deliver backlog messages to a consumer that goes offline for a period of time.
Subscription Type	Yes	<p>The following options are available:</p> <ul style="list-style-type: none"> • Exclusive: In Exclusive mode, only a single consumer is allowed to attach to the subscription. An error occurs if multiple consumers subscribe to a topic by using the same subscription. • Shared: In Shared mode, multiple consumers can attach to the same subscription. Messages are delivered in a round robin distribution across consumers, and any given message is delivered to only one consumer. • Failover: In Failover mode, multiple consumers can attach to the same subscription. • KeyShared: In KeyShared mode, multiple consumers can attach to the same subscription. Messages are delivered in a distribution across consumers and messages with the same key or same ordering key are delivered to only one consumer.
Initial Position	Yes	On the Initial Position field, select Latest to receive the messages that have been published after the subscriber has been connected. Select Earliest to receive all the stored and new messages.
DLQ Topic	No	<p>The DLQ Topic is available only if the Subscription Type is Shared. If the flow started by the trigger does not complete successfully, the message is negatively acknowledged. If this happens repeatedly, the message effectively blocks processing by the trigger. This field allows you to relocate messages that cannot be processed to another topic. If the DLQ Topic field is not provided then the dead letter queue (DLQ) processing is not performed.</p> <div style="display: flex; align-items: center;">  <div style="border-left: 1px solid #ccc; padding-left: 10px;"> <p>A delay exists between a negative acknowledgment and reposting the message. So it can take several minutes for a message to arrive on the DLQ.</p> </div> </div>
DLQ Max Deliveries	No	The number of times a message is negatively acknowledged before being rerouted to the DLQ Topic field.
Message Format	Yes	<p>The Message Format field controls the format of the output schema. The following options are available:</p> <ul style="list-style-type: none"> • String • JSON: If JSON is selected, a text editor is provided on the Output Settings tab to accept a JSON document.

Output Settings

The **Output Settings** tab has the following fields:

Field	Required	Description
Message Properties	No	You can add property value to the properties field presented on the Input schema. Each additional property is presented as a named string to be mapped.
Schema for JSON value	No	The Schema for JSON value field is only available when the Message Format field on the Triggers Settings tab is JSON. This is a free form text editor that accepts any valid JSON document, which is then presented on the output schema.

Map to Flow Inputs

The **Map to Flow Inputs** tab has the following fields:

Field	Description
payload	Either a simple string or a representation of the JSON document provided on the Output Settings tab.
properties	An object with a string value for each of the named properties from the Output Settings tab.
topic	If the subscriber subscribes to multiple topics, the topic field provides the actual topic on which the message has arrived.

Apache Pulsar Producer Activity

Apache Pulsar Producer activity can be used to map a string or a JSON object to a message. You can use the Producer activity to map the properties such as `string:stringpairs` and an optional key. You can send messages to a single topic and compress them by all the supported algorithms.

Settings

On the **Settings** tab, configure the following settings:

Field	Required	Description
Pulsar Connection	Yes	Name of the connection.
Topic Name	Yes	<p>The name of the topic to which the message is published. The Topic Name field has the following format <code>:persistent://public/default/foo</code></p> <p>The first segment is the type of topic. The topic can be persistent or non-persistent. With persistent topics, all the messages are persisted to the disk. The second segment is the name of the domain or tenant. The third segment is the namespace within the domain and the fourth segment is the name of the individual topic.</p> <p>An abbreviated form of the topic name can be used only when the right most segment is provided. If the other segments of the topic name are needed then the complete topic name must be used.</p>

Field	Required	Description
Compression Type	No	The following options are available for the compression type: <ul style="list-style-type: none"> • None • LZ4 • ZLIB • ZSTD

Input Settings

The **Input Settings** tab has the following fields:

Field	Required	Description
Message Format	Yes	The following options are available for the message format: <ul style="list-style-type: none"> • String: A simple string can be mapped on the Input schema. • JSON: The Input schema is updated to reflect the structure of the JSON document and flow data can be mapped to the fields.
Message Properties	No	You can use the Message Properties field to add property value to the properties field presented on the Input schema. Each additional property is presented as a named string that is mapped.
Schema for JSON Value	No	You can see the Schema for JSON Value field only if the Message Format on the Input Settings tab is JSON. This is a free form text editor, which accepts any valid JSON document that is presented on the Input schema. This JSON document must also be used by the receiving application to decode the message.

Input

The **Input** tab has the following fields:

Field	Description
payload	The message to be published to a topic. Either a simple string or a representation of the JSON document provided on the Input Settings tab
properties	An object with a string value for each of the named properties from the Input Settings tab
key	A string value used by the topic compaction function of the broker. In the shard subscriber mode subscribers are bound to specific keys so that the repeat keys are always processed by the same consumer

Output Settings

The **Output Settings** tab has the following field:

Field	Description
msgid	A string representing the message id.

Output

The **Output** tab displays the output schema of the activity as a tree structure. The output is read-only. The information in the schema varies based on the fields selected on the **Settings** tab. The properties that are displayed in the schema correspond to the output of this activity and can be used as input by subsequent activities in the flow.

Microsoft SQL Server

This section contains the following topics:

Overview

TIBCO Flogo[®] Connector for Microsoft SQL Server allows you to connect to Microsoft SQL Server and run SQL statement to query, insert, update, or delete data.

Creating a Microsoft SQL Server Connection

You must create a SQL Server connection before you can use the TIBCO Flogo[®] Connector for Microsoft SQL Server activities. The Microsoft SQL Server connection contains all the parameters required to connect to the Microsoft SQL Server database. The Microsoft SQL Server connection is used by all the activities in the Microsoft SQL Server category.



By default, the Microsoft SQL Server database listens for connections on port 1433, although that is configurable as port is the connection property on the Microsoft SQL Server connector.

To create a Microsoft SQL Server connection, perform the following steps:

1. Click the **Connections** tab and perform one of the following actions:
 - To add a connection for the first time, click the **SQL Server Connector** card. You can search for a connector by typing the connector name in the search field.
 - If you have existing connections and want to add a new connection, click the **Add Connection** link.
2. In the SQL Server Connector dialog box, enter the connection details. For field descriptions, see [Microsoft SQL Server Connection Details](#).
3. Click **Save Connection**.

Microsoft SQL Server Connection Details

Provide the information of the Microsoft SQL Server that this connection connects to.

The SQL Server Connector dialog box contains the following fields:

Field	Description
Name	A name for the Microsoft SQL Server connection that you are creating
Description	A short string describing the connection
Host	URL of the server that hosts the Microsoft SQL Server database

Field	Description
Port	<p>Port number on which the Microsoft SQL Server database listens</p> <p> By default the Microsoft SQL Server Connector cluster is configured with port 1433. For custom configurations the valid port range is 1024 - 32767.</p>
Database Name	Name of the Microsoft SQL Server database
User	User name of the Microsoft SQL Server database user
Password	Password for the Microsoft SQL Server database account
Maximum Open Connections	<p>Sets the maximum number of open connections to the database.</p> <p>If Maximum Idle Connections is greater than 0 and the Maximum Open Connections is less than the Maximum Idle Connections, then the Maximum Idle Connections is reduced to match the Maximum Open Connections value.</p> <p>Default: 0 (If value is less than or equal to 0, then there is no maximum limit to the number of open connections to the database.)</p>
Maximum Idle Connections	<p>Sets the maximum number of connections in the idle connection pool.</p> <p>If Maximum Open Connections is greater than 0, but less than the Maximum Idle Connections, then the Maximum Idle Connections is reduced to match the Maximum Open Connections value.</p> <p>Default: 2</p> <p>If value is less than or equal to 0, then no idle connections are retained.</p>
Maximum Connection Lifetime	<p>Sets the maximum amount of time for which a connection can be reused.</p> <p>Expired connections are closed before reuse.</p> <p>Default: 0 (If value is less than or equal to 0, then connection can be used forever.)</p> <p>Valid time units: ns, us (or μs), ms, s, m, h.</p> <p>Example: 60s, 60m, 6h45m.</p>
Secure Connection	<p>True: Establishes the secure connection by providing public key certificate when set to True.</p> <ul style="list-style-type: none"> • CA Certificate: Certificate Authority (CA) certificate. • Validate Server Certificate: Verifies CA Certificate and the server host name to ensure that it matches the Common Name in the server certificate.

- **Maximum Open Connections** must be less than the limit on the number of connections imposed by your database and infrastructure.
- Higher **Maximum Open Connections** and **Maximum Idle Connections** values can lead to better performance. However, having a large idle connection pool (with connections that are not reused) can lead to reduced performance.
- To mitigate the risk mentioned in the preceding point, set a relatively short **Maximum Connection Lifetime**. However, it must not be so short that leads to connections being ended and re-created unnecessarily often.

SQLServer Query

Use this activity to execute a simple or a complex SQL Query on a Microsoft SQL Server database. The **SQLServer Query** activity returns information in the form of rows.

Settings

The **Settings** tab has the following fields:

Field	Description
Connection	Name of the Microsoft SQL Server database connection from where information is retrieved.

Input Settings

Field	Description
Query	<p>An SQL statement used to query the database. The query can be a simple query or a complex query. A complex query has nested SQL statements. Prepared SQL queries can be constructed by using substitution variables (or substitution parameters) of the form ?<fieldname> in the query statement. For example <code>select lastname, firstname, title from employees where lastname=? lastname'</code>;</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at runtime. Input and Output tabs are populated based on the query.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>Following are some examples of simple and complex queries:</p> <ul style="list-style-type: none"> Simple query example: <pre>SELECT * FROM employees;</pre> <p>For the above query, the output fields are generated from the columns of the table employees.</p> <pre>SELECT lastname, firstname FROM employees WHERE country = ?country and lastname like '%11%' ORDER BY lastname DESC, firstname ASC;</pre> <p>For the above query, output fields are generated for lastname and firstname and input fields are generated for country. Also the mapped value for the field country is substituted into the substitution variable ? country at run time.</p> Nested query example: <pre>SELECT categoryID, productName, MAX(unitprice) FROM products A WHERE unitprice = (SELECT MAX(unitprice) FROM products B WHERE B.categoryID = A.categoryID) GROUP BY categoryID, productName HAVING MAX(unitprice) > 100;</pre>
Manually Configure Metadata	Set the Manually Configure Metadata field to True to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	The grid is provided for informational purposes only.

Input

This tab contains the input schema. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code their values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Output

This tab displays the activity output schema in a tree structure format. The output of an activity is displayed for informational purposes only and cannot be modified or altered. The information in this schema varies depending on the fields that you selected on the **Input Settings** tab.

The properties that are displayed on the **Output** tab schema correspond to the output of this activity and can be used as input by subsequent activities in the flow.

SQLServer Insert

Use this activity to execute an SQL Insert to insert the records into the Microsoft SQL Server database.

Settings

The **Settings** tab has the following fields.

Field	Description
Connection	Name of the Microsoft SQL Server database connection from where information is retrieved. You can select a connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Insert	<p>An SQL statement used to insert a record in the table. You can construct prepared SQL queries by using substitution variables (or parameters) of the form ?<fieldname> in the insert query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at run time. If the query contains parameters then only parameter metadata is populated.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query. When substitution variables used in the VALUES clause are identical to the column names in the table, those variables are displayed under the Values array and otherwise displayed under Parameters on the Input tab. Use Values to insert multiple records at once and use Parameters to insert only one record at a time.</p> <p>Map JSON data from previous activity for insert when you use:</p> <ul style="list-style-type: none"> • Values - https://github.com/TIBCOSoftware/tci-flogo/tree/master/samples/app-dev/array.forEach.sample • Parameters - https://github.com/TIBCOSoftware/tci-flogo/tree/master/samples/app-dev/loops.sample <p>For more details about Data Mappings, see "Mapping JSON Data with the json.path() Function" section in <i>TIBCO Flogo Enterprise User's Guide</i>.</p> <p>The following example represents a typical insert query:</p> <pre>INSERT INTO ADVISOR (s_id, i_id) VALUES (?s_id, ?i_id);</pre> <p>For the above insert query, there will not be any output field, and input field is generated for s_id (VARCHAR) and i_id (VARCHAR) under Values[] node as its part of the values clause. Also, the mapped value for the field price and name is substituted into the substitution variable ?s_id and ?i_id.</p>
Manually Configure Metadata	Set the Manually Configure Metadata field to True to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	The grid is provided for informational purpose only.

Input

This tab displays the input schema of the activity in a tree structure format. The information in the schema varies based on the insert query provided. Substitution parameters in the query, which exactly match column names, are presented in the Values node and substitutions, which do not match, appear in the Parameters node.

Output

This tab displays the output schema of the activity in a tree structure format. The output tab displays rowsAffected and lastInsertId. rowsAffected equals the number of rows inserted and lastInsertId will be set to the id of the last server generated unique key value if the table is defined that way, otherwise -1.

SQLServer Update

Use this activity to execute an update statement on one or more rows of an Microsoft SQL Server table.

Configuration

The **Configuration** tab has the following fields.

Field	Description
Connection	Name of the Microsoft SQL Server database connection from where information is retrieved. You can select a connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Update Statement	<p>An SQL statement used to update one or more rows in a table. You can construct prepared SQL queries by using substitution parameters of the form ? <fieldname> in the query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted at run time. If the query contains parameters then only parameter metadata is populated.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>The following example represents a typical update query:</p> <pre>UPDATE pet SET Name=?Name , Color = ?Color WHERE Species = 'cat';</pre> <p>For the above query, all rows in the 'pet' table will have their Name and Color columns set to the values provided on the Input tab where the species is cat. As usual all rows satisfying the where clause will be updated. If the where clause is omitted then ALL rows are updated. The limit clause can also be used to control this behavior.</p> <p>It is also possible to use a subquery to derive the values to be used in the update. The subquery format follows the general form:</p> <pre>UPDATE table_name SET column1 = ?column1, column2 = ?column2..., columnN = ? columnN WHERE [condition];</pre> <p>In this case the parameters supplied are used in the select query that provides replacement values in the outer update statement. The where clause could also be parameterized.</p>
Manually Configure Metadata	Set the Manually Configure Metadata field to True to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	The grid is provided for informational purpose only and documents the fields that are made available as parameters on the input tab.

Input

This tab displays the input schema of the activity in a tree structure format. The information in the schema varies based on the update query provided. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code the values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Substitution parameters from the query statement are provided in the Parameters node.

Output

The Update activity returns the number of rows affected by the query.

SQLServer Delete

Use this activity to execute an SQL Delete to delete the record based on the delete statement.

Settings

The **Settings** tab has the following fields:

Field	Description
Connection	Name of the Microsoft SQL Server database connection from where information is retrieved. You can select the connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Delete	<p>A SQL statement used to delete the record from the table. You can construct prepared SQL queries by using substitution variables (or parameters) of the form <code>?<fieldname></code> in the delete query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at runtime. Input and Output tabs are populated based on the query.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>The following example represents a typical delete query:</p> <pre>Delete from advisor where i_id=?i_id;</pre> <p>For the above delete query, there will be no output field and input field is generated for i_id (VARCHAR). Also the mapped value for the field name is substituted into the substitution variable ?i_id.</p>
Manually Configure Metadata	Set the Manually Configure Metadata field to <code>True</code> to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	The grid is provided for informational purpose only.

Input

This tab displays the input schema of the activity in a tree structure format. The information in the schema varies based on the delete query provided. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code the values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Output

The Output tab displays the output schema of the activity in a tree structure format. The output is read-only. The output tab rowsAffected field is set to the number of rows deleted by the activity.

Manually Configuring Metadata

You can use **Manually Configure Metadata** option to fetch table metadata manually.

When the field is set to `True`, one more **Query** field at the bottom of this option is displayed. To retrieve and configure the database schema fields, you must first enter a simple schema query in this field and click **Fetch**. For example, you can enter the following schema query to fetch the metadata:

```
SELECT * FROM CUSTOMERS;
```

After the successful processing of the query, the table field at the bottom is populated with the column metadata information. You can add, edit, and remove fields in the table. The fields marked **Selected** in the table are available on the **Output** tab of the activity. The fields selected in the **Parameter** column are available on the **Input** tab of the activity. The fields selected in the **Value** column are available on the **Output** tab of the activity. You can modify these settings by selecting or clearing the check boxes.

MQTT

This section contains the following topics:

Overview

TIBCO Flogo[®] Connector for MQTT provides out-of-the-box connectivity for MQTT to subscribe to a topic or publish messages to a topic. It supports security and authentication, quality of service, and message retention.



For information about how to use MQTT, see the MQTT documentation.

Creating the MQTT Connection

To use this connector, you must first create an MQTT connection. The MQTT connection contains the parameters required to connect to the MQTT server. The MQTT connection is used by all the activities in the MQTT category.

The MQTT Publisher activity and MQTT Subscriber trigger reconnect automatically to the MQTT broker in an event of a connection failure. The client uses a **cleanSession** flag value of `true`. If the client is disconnected for any reason, all the messages from a previous session are lost.

Prerequisites

Before you create a connection, familiarize yourself with MQTT. For details about how to use the product, see MQTT documentation.

Procedure

1. On the global navigation bar, click **Connections** and perform one of the following actions:

- To add a connection for the first time, click the **MQTT Connector** card. You can search for a connector by typing the connector name in the search field.
 - If you have an existing connection and want to add a new connection, click **Create**.
2. In the MQTT Connector dialog box, enter the connection details. For field descriptions, see the [MQTT Connection Details](#) topic.
 3. Click **Save**.

MQTT Connection Details

To establish the connection successfully, you must configure the MQTT instance.

The MQTT Connector dialog box contains the following fields:

Field	Description
Connection Name	A unique name for the connection that you are creating. This is displayed in the Connection drop-down list for all the TIBCO Flogo [®] Connector for MQTT activities.
Description	A short description of the connection
Broker URL	The format of the broker URL is: <proto>://<host>:<port> Where proto is either "tcp" or "ssl".
Username	Enter the user name required to authenticate the broker port for this connection.
Password	Enter the password required to authenticate the broker port for this connection.
Encryption Mode	Choose the encryption mode from None, TLS-Cert, or TLS-ClientAuth.
CA Certificate	Browse to the location of the certificate authority(CA) certificate used to sign the certificate of the server. The content of the file is included in the configuration of the application. For information about using an application property for this field, see Rules for Application Property .
Client Certificate	Browse to the location of the certificate file of the client. The content of the file is included in the configuration of the application. This option is available only if the encryption mode is set to "Client Auth". For information about using an application property for this field, see Rules for Application Property .
Client Key	Browse to the location of the secret key of the client and select it. This option is available only if the encryption mode is set to "Client Auth". For information about using an application property for this field, see Rules for Application Property .

Rules for Application Property

The following rules must be followed when an application property is used for any of the certificate or key fields:

- A base64 encoded privacy-enhanced mail(PEM) string representing the certificate or the key must be entered in the following format:
base64 , <base64content>

This format is included in the configuration of the application and is available wherever the application is deployed.

- A file by using the Uniform Resource Identifier(URI) syntax in the following format:

```
file:///somedir/somefile.PEM
```

This file name is stored in the configuration of the application. The file must be available by this name wherever the application is deployed.

- A file by using a fully qualified name in the following format:

```
/somedir/somefile.PEM
```

This file name is stored in the configuration of the application. The file must be available by this name wherever the application is deployed.

MQTT Subscriber Trigger

The MQTT Subscriber Trigger subscribes to a topic and presents the messages received as output. When the messages arrive, a new flow is triggered. The MQTT Subscriber Trigger supports multiple trigger handlers. A single MQTT Subscriber trigger listens to multiple topics and in turn executes multiple flows.

Settings

The **Settings** tab has the following fields:

Field	Description
Trigger Settings	The trigger settings are common to the trigger across all flows that use the trigger.
Handler Settings	The handler settings are applicable to a specific flow attached to the trigger.
Maximum QoS	If a message is sent with a higher QoS, the effective QoS of the message is reduced to the maximum QoS value. To avoid QoS getting reduced on the subscriber side, you must set maximum QoS to 2. The QoS reported on the output schema of the trigger is a copy of this setting and not the QoS established by the publisher.
Value Deserializer	<p>Establish the way the message body is treated</p>  <p>Messages received by the subscriber must conform to the value deseriaization scheme configured for the subscriber. An error occurs if the subscriber receives a message that does not conform to the configured deserialization scheme and schema.</p>
Value Deserializer:String	The activity input presents the message bytes as a simple string
Value Deserializer:JSON	The activity input presents the message bytes as a JSON object decoded with the provided schema
Value Deserializer:Base64	The activity input presents the message bytes as a base64 encoded string

Field	Description
Show Will Fields	<p>When set to true, you can use Show Will Fields to view the last will and testament fields. When Show Will Fields is set to true, it populates the following fields:</p> <ul style="list-style-type: none"> • Will: Enter the message to be sent to the WillTopic when the connection terminates abnormally. • Will Topic: Enter a topic to which the Will message is sent when there is a connection failure. • Will QoS: Quality of Service for the Will message. Valid values are as follows: <ul style="list-style-type: none"> 0: Message is delivered at most once 1: Message is delivered at least once 2: Message is delivered exactly once • Will Retain: Set to true to allow the server to retain the Will message.

Output Settings

The **Output Settings** tab has the following field:

Field	Description
Schema for JSON value	<p>Enter a JSON object representation used on the output tab.</p> <p>For instance:</p> <pre>{ "fname": "Sam", "lname": "Patricks", "age": 37, "employed": true }</pre>

Output

The **Output** tab displays the schema in a tree format. The output is read-only.

The **Output** tab has the following fields:

Field	Description
topic	The exact topic on which the message arrived
retained	It indicates whether this message was a retained message
qos	It indicates the quality of service of the message.
duplicate	It is set to true when the message is received, but not acknowledged. The message is then re-received. This happens in the event of a subscriber failure or uncontrolled application shutdown.

Field	Description
messageID	It is a sequential integer enumerating all the messages processed by the subscriber because it is instantiated.
string value	The value of the message presented as a string. An error is displayed if the message is not a string.

MQTT Publish Activity

The MQTT Publish activity publishes a message to a broker, which supports and exposes the MQTT protocol.

Settings

The **Settings** tab has the following fields:

Field	Description
Connection	Name of the connection
Topic	The topic where the message is published
Retain	You must enable this field to retain the last message for new subscribers
QoS	The quality of service is set to 0, 1, or 2 <ul style="list-style-type: none"> • 0 - message is delivered at most once • 1 - message is delivered at least once • 2 - message is delivered exactly once
Value Deserializer	Establish the way the message body is treated
Value Deserializer:String	The activity input accepts a simple string. The string is sent as an array of bytes to the server.
Value Deserializer:JSON	An Input Settings field is enabled where the application designer can enter a JSON object. That object is presented on the input schema for mapping and at run time the object is deserialized and sent as bytes. A subscriber can use the same JSON object definition to deserialize the message.
Value Deserializer:Base64	The activity accepts string data which is base64 decoded into a byte array before being sent to the broker. If a previous activity presents a byte array on its output, it is safe to map that value here.

Input Settings

The **Input Settings** tab is visible if the deserializer is set to JSON on the **Settings** tab. An example of a JSON object is entered in this field which can be used to compose the input schema.

The **Input Settings** tab has the following field:

Field	Description
Schema for JSON value	<p>Enter a JSON object representation to be used on the Input tab.</p> <p>Sample JSON query:</p> <pre>{ "fname": "Sam", "lname": "Patricks", "age": 37, "employed": true }</pre>

Input

The **Input** tab has the following field:

Field	Description
Input	Map the string, byte array, or JSON object.

Loop

If you want this activity to iterate multiple times within the flow, enter an expression that evaluates to the iteration details. Select a type of iteration from the Type menu. The default type is None, which means the activity does not iterate. For more information, refer to the "Using the Loop Feature in an Activity" topic in the TIBCO Flogo® Enterprise documentation.

Oracle MySQL

This section contains the following topics:

Overview

TIBCO Flogo® Connector for Oracle MySQL allows you to connect to a Oracle MySQL database and run SQL statement to query, insert, update or delete data.

Creating an Oracle MySQL Connection

You must create an Oracle MySQL connection before you can use the Oracle MySQL activities. The Oracle MySQL connection contains all the parameters required to connect to the Oracle MySQL database. The Oracle MySQL connection is used by all the activities in the Oracle MySQL category.



By default, the Oracle MySQL database listens for connections on port 3306, although that is configurable as port is the connection property on the MySQL Connector.

To create an Oracle MySQL connection, perform the following steps:

1. Click the **Connections** tab and perform one of the following actions:
 - To add a connection for the first time, click the **MySQL Connector** card. You can search for a connector by typing the connector name in the search field.
 - If you have existing connections and want to add a new connection, click the **Add Connection** link.

2. In the MySQL Connector dialog box, enter the connection details. For field descriptions, see [Oracle MySQL Connection Details](#).
3. Click **Save Connection**.

Oracle MySQL Connection Details

Provide the information for the Oracle MySQL server that this connection connects to.

The MySQL Connector dialog box contains the following fields:

Field	Description
Name	A name for the Oracle MySQL connection that you are creating
Description	A short string describing the connection
Host	URL of the server that hosts the Oracle MySQL database
Port	Port number on which the Oracle MySQL database listens  By default the Oracle MySQL Connector cluster is configured with port 3306. For custom configurations, the valid port range is 1024 - 32767.
Database Name	Name of the Oracle MySQL database
User	User name of the Oracle MySQL database user
Password	Password for the Oracle MySQL database account
Maximum Open Connections	Sets the maximum number of open connections to the database. If Maximum Idle Connections is greater than 0 and the Maximum Open Connections is less than the Maximum Idle Connections, then the Maximum Idle Connections is reduced to match the Maximum Open Connections value. Default: 0 (If value is less than or equal to 0, then there is no maximum limit to the number of open connections to the database.)
Maximum Idle Connections	Sets the maximum number of connections in the idle connection pool. If Maximum Open Connections is greater than 0, but less than the Maximum Idle Connections, then the Maximum Idle Connections is reduced to match the Maximum Open Connections value. Default: 2 If value is less than or equal to 0, then no idle connections are retained.
Maximum Connection Lifetime	Sets the maximum amount of time for which a connection can be reused. Expired connections are closed before reuse. Default: 0 (If value is less than or equal to 0, then connection can be used forever.) Valid time units: ns, us (or μ s), ms, s, m, h. Example: 60s, 60m, 6h45m.

Field	Description
Secure Connection	Establishes the secure connection when set to True . <ul style="list-style-type: none"> • TLS: Select the TLS mode from the list. See TLS Modes.



- **Maximum Open Connections** must be less than the limit on the number of connections imposed by your database and infrastructure.
- Higher **Maximum Open Connections** and **Maximum Idle Connections** values can lead to better performance. However, having a large idle connection pool (with connections that are not reused) can lead to reduced performance.
- To mitigate the risk mentioned in the preceding point, set a relatively short **Maximum Connection Lifetime**. However, it must not be so short that leads to connections being ended and re-created unnecessarily often.

TLS Modes

Provide the following certificates for each TLS mode:

- **CA Certificate**: Certificate Authority (CA) certificate.
- **Client Certificate**: Not required. Client Certificate file for connections requiring client authentication.
- **Client Key**: Not required. Client key file used for connections requiring client authentication.

For more details on TLS modes, see [MySQL Documentation](#).

TLS Modes

TLS Modes	Description
Required	Client requires an encrypted connection and fails if one cannot be established.
Preferred	If an attempt to connect to an encrypted connection fails, then the unencrypted connection is established.
VerifyCA	Client require an encrypted connection, and verifies with the server CA certificate. <ul style="list-style-type: none"> • Validate Server Certificate: Verifies CA certificate and the server host name to ensure it matches the Common Name in the server certificate.
VerifyIdentity	Server host name is verified to ensure that it matches the host name stored in the server certificate.

MySQL Query

Use this activity to execute a simple or a complex SQL Query on an Oracle MySQL database. The **MySQL Query** activity returns information in the form of rows.

Configuration

The **Configuration** tab has the following fields:

Field	Description
Connection	Name of the Oracle MySQL database connection from where information is retrieved

Input Settings

Field	Description
Query	<p>An SQL statement used to query the database. The query can be a simple query or a complex query. A complex query has nested SQL statements. Prepared SQL queries can be constructed by using substitution variables (or substitution parameters) of the form ?<fieldname> in the query statement. For example <code>select * from classroom where building like ?building;</code></p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at runtime. Input and Output tabs are populated based on the query.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>Following are some examples of simple and complex queries:</p> <ul style="list-style-type: none"> Simple query example: <pre>SELECT * FROM classroom;</pre> <p>For the above query, the output fields are generated from the columns of the table <code>classroom</code>.</p> <pre>SELECT name, dept_name, tot_cred FROM university.student WHERE dept_name = ?dept_name and tot_cred >= ?tot_cred order by tot_cred;</pre> <p>For the above query, output fields are generated for <code>name</code>, <code>dept_name</code> and <code>tot_cred</code> and input fields are generated for <code>dept_name</code> and <code>tot_cred</code>. Also the mapped values for the fields <code>dept_name</code> and <code>tot_cred</code> are substituted into the substitution variables <code>?dept_name</code> and <code>?tot_cred</code> at run time.</p> Nested query example: <pre>SELECT firstname, lastname, total_quantity FROM (SELECT buyerid, sum(qtysold) total_quantity FROM sales GROUP BY buyerid ORDER BY total_quantity desc limit 10) Q, users WHERE Q.buyerid = userid ORDER BY Q.total_quantity desc;</pre>
Manually Configure Metadata	Set the Manually Configure Metadata field to <code>True</code> to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	Displays the database column metadata.

Input

This tab contains the input schema. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code their values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Output

This tab displays the activity output schema in a tree structure format. The output of an activity is displayed for informational purposes only and cannot be modified or altered. The information in this schema varies depending on the fields that you selected on the **Input Settings** tab.

The properties that are displayed on the **Output** tab schema correspond to the output of this activity and can be used as input by subsequent activities in the flow.

MySQL Insert

Use this activity to execute an SQL Insert to insert the records into the Oracle MySQL database.

Configuration

The **Configuration** tab has the following fields.

Field	Description
Connection	Name of the Oracle MySQL database connection from where information is retrieved. You can select a connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Insert Statement	<p>An SQL statement used to insert a record in the table. You can construct prepared SQL queries by using substitution variables (or parameters) of the form ?<fieldname> in the insert query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at run time. If the query contains parameters then only parameter metadata is populated.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query. When substitution variables used in the VALUES clause are identical to the column names in the table, those variables are displayed under the Values array and otherwise displayed under Parameters on the Input tab. Use Values to insert multiple records at once and use Parameters to insert only one record at a time.</p> <p>Map JSON data from previous activity for insert when you use:</p> <ul style="list-style-type: none"> • Values - https://github.com/TIBCOSoftware/tci-flogo/tree/master/samples/app-dev/array.forEach.sample • Parameters - https://github.com/TIBCOSoftware/tci-flogo/tree/master/samples/app-dev/loops.sample <p>For more details about Data Mappings, see "Mapping JSON Data with the json.path() Function" section in <i>TIBCO Flogo Enterprise User's Guide</i>.</p> <p>The following example represents a typical insert query:</p> <pre>INSERT INTO ADVISOR (s_id, i_id) VALUES (?s_id, ?i_id);</pre> <p>For the above insert query, there will not be any output field, and input field is generated for s_id (VARCHAR) and i_id (VARCHAR) under Values[] node as its part of the values clause. Also, the mapped value for the field price and name is substituted into the substitution variable ?s_id and ?i_id.</p> <p>It is also possible to enter multi-row queries. For example:</p> <pre>INSERT INTO products (product_no, name, price) VALUES (1, 'Cheese', 9.99), (2, 'Bread', 1.99), (3, 'Milk', 2.99)</pre> <p>Value substitution variables can also be used in multi-row values. For example:</p> <pre>INSERT INTO products VALUES (1, 'Cheese', ?price), (2, 'Bread', ?price), (3, 'Milk', ?price)</pre> <p>If multi-row values are used, it is important to note the following points:</p> <ol style="list-style-type: none"> 1. A value substitution variable must appear in all the rows at the same position. You cannot have some rows with the value variable and other rows without it. 2. If the value row is mapped to the output values from previous activities, then the output data from the previous activity must contain the same number of records as there are rows entered in the insert query definition. 3. For multiple record insertion, mixing of value substitution variables and parameter substitution variables in each record entry is not recommended. Either they must all be value substitution variables or they must be all

Field	Description
	parameter substitution variables. They should not be a combination of both. For example, the following use is discouraged: <pre>INSERT INTO products (ProductNo, Name, Price) VALUES (?ProductNo, ?Name, ?Price), (?myproduct, ?myname, ? myprice);</pre>
Manually Configure Metadata	Set the Manually Configure Metadata field to True to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	Displays the database column metadata.

Input

This tab displays the input schema of the activity in a tree structure format. The information in the schema varies based on the insert query provided. Substitution parameters in the query, which exactly match column names, are presented in the Values node and substitutions, which do not match, appear in the Parameters node.

Output

This tab displays the output schema of the activity in a tree structure format. The output tab displays rowsAffected and lastInsertId. rowsAffected equals the number of rows inserted and lastInsertId is set to the ID of the last server generated unique key value if the table is defined that way, otherwise -1.

MySQL Update

Use this activity to execute an update statement on one or more rows of an Oracle MySQL table.

Configuration

The **Configuration** tab has the following fields.

Field	Description
Connection	Name of the Oracle MySQL database connection from where information is retrieved. You can select a connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Update Statement	<p>An SQL statement used to update one or more rows in a table. You can construct prepared SQL queries by using substitution parameters of the form ? <fieldname> in the query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted at run time. If the query contains parameters then only parameter metadata is populated.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>The following example represents a typical update query:</p> <pre>UPDATE pet SET Name=?Name , Color = ?Color WHERE Species = 'cat';</pre> <p>For the above query all rows in the 'pet' table will have their Name and Color columns set to the values provided on the Input tab where the species is cat. As usual all rows satisfying the where clause will be updated. If the where clause is omitted then ALL rows are updated. The limit clause can also be used to control this behavior.</p> <p>It is also possible to use a subquery to derive the values to be used in the update. The subquery format follows the general form:</p> <pre>UPDATE table_name SET column1 = ?column1, column2 = ?column2..., columnN = ? columnN WHERE [condition];</pre> <p>In this case the parameters supplied are used in the select query that provides replacement values in the outer update statement. The where clause could also be parameterized.</p>
Manually Configure Metadata	Set the Manually Configure Metadata field to True to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	Displays the database column metadata.

Input

This tab displays the input schema of the activity in a tree structure format. The information in the schema varies based on the update query provided. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code the values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Substitution parameters from the query statement are provided in the Parameters node.

Output

The Update activity returns the number of rows affected by the query.

MySQL Delete

Use this activity to run an SQL Delete to delete the record based on the delete statement.

Configuration

The **Configuration** tab has the following fields:

Field	Description
Connection	Name of the Oracle MySQL database connection from where information is retrieved. You can select the connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Delete Statement	<p>A SQL statement used to delete the record from the table. You can construct prepared SQL queries by using substitution variables (or parameters) of the form <code>?<fieldname></code> in the delete query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at runtime. Input and Output tabs are populated based on the query.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>The following example represents a typical delete query:</p> <pre>Delete from advisor where i_id=?i_id;</pre> <p>For the above delete query, there is no output field and input field is generated for <code>i_id</code> (VARCHAR). Also the mapped value for the field name is substituted into the substitution variable <code>?i_id</code>.</p>
Manually Configure Metadata	Set the Manually Configure Metadata field to <code>True</code> to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	Displays the database column metadata.

Input

This tab displays the input schema of the activity in a tree structure format. The information in the schema varies based on the delete query provided. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code the values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Output

The Delete activity returns the number of rows affected by the query.

Manually Configuring Metadata

You can use **Manually Configure Metadata** option to fetch table metadata manually.

When the field is set to `True`, one more **Query** field at the bottom of this option is displayed. To retrieve and configure the database schema fields, you must first enter a simple schema query in this field and click **Fetch**. For example, you can enter the following schema query to fetch the metadata:

```
SELECT * FROM CUSTOMERS;
```

After the successful processing of the query, the table field at the bottom is populated with the column metadata information. You can add, edit, and remove fields in the table. The fields marked as **Selected** in the table are available on the **Output** tab of the activity. The fields selected in the **Parameter** column are

available on the **Input** tab of the activity. The fields selected in the **Value** column are available on the **Output** tab of the activity. You can modify these settings by selecting or clearing the check boxes.

PostgreSQL

This section contains the following topics:

Overview

TIBCO Cloud™ Integration - Flogo® (PAYG) enables you to run SQL queries on a PostgreSQL or a Greenplum database instance. A PostgreSQL connection is used to create a PostgreSQL or Greenplum query activity.

For details about using PostgreSQL or Greenplum Database, see their respective product documentation.

Creating a PostgreSQL Connection

You must create a PostgreSQL connection before you can use this connector. The PostgreSQL connection contains all the parameters required to connect to the PostgreSQL database. This connection is used by all the activities in the PostgreSQL category.



By default, the PostgreSQL database listens for connections on port 5432, although that is configurable as port is the connection property on the PostgreSQL Connector.

To create a PostgreSQL connection, perform the following steps:

1. Click the **Connections** tab and perform one of the following actions:
 - To add a connection for the first time, click the **PostgreSQL Connector** card. You can search for a connector by typing the connector name in the search field.
 - If you have existing connections and want to add a new connection, click the **Add Connection** link.
2. In the PostgreSQL Connector dialog box, enter the connection details. For field descriptions, see [PostgreSQL Connection Details](#).
3. Click **Save Connection**.

PostgreSQL Connection Details

Provide the information of the PostgreSQL server that this connection can connect to.

The PostgreSQL Connector dialog box contains the following fields:

Field	Description
Connection Name	A name for the PostgreSQL connection that you are creating
Description	A short string describing the connection
Host	URL of the server that hosts the PostgreSQL database
Port	Port number on which the PostgreSQL Connector database listens  By default PostgreSQL Connector cluster is configured with port 5432. For custom configurations, the valid port range is 1024 - 32767.
Database Name	Name of the PostgreSQL database

Field	Description
User	User name of the PostgreSQL database user
Password	Password for the PostgreSQL database account
Maximum Open Connections	<p>Sets the maximum number of open connections to the database.</p> <p>If Maximum Idle Connections is greater than 0 and the Maximum Open Connections is less than the Maximum Idle Connections, then the Maximum Idle Connections is reduced to match the Maximum Open Connections value.</p> <p>Default: 0 (If value is less than or equal to 0, then there is no maximum limit to the number of open connections to the database.)</p>
Maximum Idle Connections	<p>Sets the maximum number of connections in the idle connection pool.</p> <p>If Maximum Open Connections is greater than 0, but less than the Maximum Idle Connections, then the Maximum Idle Connections is reduced to match the Maximum Open Connections value.</p> <p>Default: 2</p> <p>If value is less than or equal to 0, then no idle connections are retained.</p>
Maximum Connection Lifetime	<p>Sets the maximum amount of time for which a connection can be reused.</p> <p>Expired connections are closed before reuse.</p> <p>Default: 0 (If value is less than or equal to 0, then connection can be used forever.)</p> <p>Valid time units: ns, us (or μs), ms, s, m, h.</p> <p>Example: 60s, 60m, 6h45m.</p>
Secure Connection	<p>Establishes the secure connection when set to True.</p> <ul style="list-style-type: none"> • TLS: Select the TLS mode from the list. See TLS Connection Modes.

- 
- **Maximum Open Connections** must be less than the limit on the number of connections imposed by your database and infrastructure.
 - Higher **Maximum Open Connections** and **Maximum Idle Connections** values can lead to better performance. However, having a large idle connection pool (with connections that are not reused) can lead to reduced performance.
 - To mitigate the risk mentioned in the preceding point, set a relatively short **Maximum Connection Lifetime**. However, it must not be so short that leads to connections being ended and re-created unnecessarily often.

TLS Modes

TLS Modes	Description
VerifyCA	<p>Verifies Certificate Authority (CA) certificate.</p> <ul style="list-style-type: none"> • CA Certificate: Not Required. Provide CA certificate for verification. • Client Certificate: Not required. Client certificate file for connections requiring client authentication. • Client Key: Not required. Client key file used for connections requiring client authentication.
VerifyFull	<p>The server host name is verified to ensure that it matches the Common Name in the server certificate.</p> <ul style="list-style-type: none"> • CA Certificate: Required. Provide CA certificate for verification. • Client Certificate: Not required. Client certificate file for connections requiring client authentication. • Client Key: Not required. Client key file used for connections requiring client authentication.



The Client Key file permissions must be 0660.

For more details on TLS modes, see [PostgreSQL SSL Support Documentation](#).

PostgreSQL Query

Use this activity to run a simple or a complex SQL Query on a database. The **PostgreSQL Query** activity returns information in the form of rows.

Settings

The **Settings** tab has the following field:

Field	Description
Connection	Name of the PostgreSQL database connection from where information is retrieved. You can select the connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Query	<p>An SQL statement used to query the database. The query can be a simple query or a complex query. A complex query has nested SQL statements. You can construct prepared SQL queries by using substitution variables (or parameters) of the form <code>?<fieldname></code> in the query statement. For example, <code>select * from student where name = ?name;</code></p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at runtime. Input and Output tabs are populated based on the query.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>The following examples represent simple and complex queries:</p> <ul style="list-style-type: none"> Simple query example: <pre>SELECT * FROM student;</pre> <p>For the above query, the output fields are generated from the column information of the table student.</p> <pre>SELECT name, dept_name, tot_cred FROM student WHERE dept_name = ?dept_name and tot_cred > ?tot_cred ORDER BY dept_name;</pre> <p>For the above query, output fields are generated for name, dept_name and tot_cred and input fields are generated for dept_name (varchar) and tot_cred (numeric). Also the mapped values for the fields, dept_name and tot_cred are substituted into the substitution variables ?dept_name and ?tot_cred at run time.</p> Nested query example: <pre>SELECT * FROM (SELECT dept_name, SUM(tot_cred) AS total_credit FROM student GROUP BY dept_name) SUBS, department WHERE SUBS.dept_name = department.dept_name AND total_credit > 8000;</pre>
Manually Configure Metadata	Set the Manually Configure Metadata field to True to fetch table metadata manually. For more information, see Manually Configure Metadata .
Fields	The grid is provided for informational purposes only.

Input

This tab contains the input schema. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code their values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Output

This tab displays the activity output schema in a tree structure format. The output of an activity is displayed for informational purposes only and cannot be modified or altered. The information in this schema varies depending on the fields that you selected on the **Input Settings** tab.

The properties that are displayed on the **Output** tab schema correspond to the output of this activity and can be used as input by subsequent activities in the flow.

PostgreSQL Insert

Use this activity to execute an SQL Insert to insert the records into the database and return the information based on the returning clause specified in the insert query.

Settings

The **Settings** tab has the following fields:

Field	Description
Connection	Name of the PostgreSQL database connection from where information is retrieved. You can select the connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Insert	<p>An SQL statement used to insert a record in the table. You can construct prepared SQL queries by using substitution variables (or parameters) of the form ?<fieldname> in the query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at run time. If the query contains parameters then only parameter metadata is populated.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query. When substitution variables used in the VALUES clause are identical to the column names in the table, those variables are displayed under the Values array and otherwise displayed under Parameters on the Input tab. Use Values to insert multiple records at once and use Parameters to insert only one record at a time.</p> <p>Map JSON data from previous activity for insert when you use:</p> <ul style="list-style-type: none"> • Values - https://github.com/TIBCOSoftware/tci-flogo/tree/master/samples/app-dev/array.forEach.sample • Parameters - https://github.com/TIBCOSoftware/tci-flogo/tree/master/samples/app-dev/loops.sample <p>For more details about Data Mappings, see "Mapping JSON Data with the json.path() Function" section in <i>TIBCO Flogo Enterprise User's Guide</i>.</p> <p>The following examples represent insert queries:</p> <pre>INSERT INTO products (product_no, name, price) VALUES (1, 'Cheese', ?price), (2, 'Juice', ?price), (3, 'Milk', ?price) returning (select name from instructor where name = ?name);</pre> <p>For the above insert query, output field is generated for name and input field is generated for price (NUMERIC) under Values[] node as its part of values clause, and name(VARCHAR) under parameters node as it is part of the parameter select sub-query . Also, the mapped value for the field price and name is substituted into the substitution variable ?price and ?name.</p> <pre>INSERT INTO products (product_no, name, price) VALUES (? product_no, ?name, ?price) returning price;</pre> <p>For the above insert query, output field is generated for price and input fields are generated for product_no (INTEGER), name(TEXT), and price(NUMERIC). Also, the mapped value for the field product_no, name, price is substituted into the substitution variables ?product_no, ?name, ?price. The parameters node on the Input tab will not have mappings as there is no parameter in the insert query statement.</p>
Manually Configure Metadata	Set the Manually Configure Metadata field to True to fetch table metadata manually. For more information, see Manually Configure Metadata .
Fields	The grid is provided for informational purposes only.

Input

This tab displays the input schema of the activity as a tree structure. The information in the schema varies based on the insert query provided. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code the values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Fields from RETURNING clause are displayed under Parameters node and fields from VALUES clause are displayed under VALUES node in input schema.

Output

This tab displays the output schema of the activity as a tree structure. The output is read-only. The information in the schema varies based on the fields selected on the **Input Settings** tab. The properties that are displayed in the schema correspond to the output of this activity and can be used as input by subsequent activities in the flow.

PostgreSQL Update

Use this activity to execute an update statement on one or more rows of a PostgreSQL table.

Configuration

The **Configuration** tab has the following fields.

Field	Description
Connection	Name of the PostgreSQL database connection from where information is retrieved. You can select the connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Update Statement	<p>An SQL statement used to update one or more rows in a table. You can construct prepared SQL queries by using substitution variables (or parameters) of the form ?<fieldname> in the query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted at run time. If the query contains parameters then only parameter metadata is populated.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>The following example represents a typical update query:</p> <pre>UPDATE pet SET Name=?Name , Color = ?Color WHERE Species = 'cat' ;</pre> <p>For the above query, all rows in the 'pet' table will have their Name and Color columns set to the values provided on the Input tab where the species is cat. As usual all rows satisfying the where clause will be updated. If the where clause is omitted then ALL rows are updated. The limit clause can also be used to control this behavior.</p> <p>It is also possible to use a subquery to derive the values to be used in the update. The subquery format follows the general form:</p> <pre>UPDATE table_name SET column1 = ?column1, column2 = ?column2..., columnN = ? columnN WHERE [condition];</pre> <p>In this case, the parameters supplied are used in the select query that provides replacement values in the outer update statement. The where clause could also be parameterized.</p>

Field	Description
Manually Configure Metadata	Set the Manually Configure Metadata field to <code>True</code> to fetch table metadata manually. For more information, see Manually Configuring Metadata .
Fields	The grid is provided for informational purpose only and documents the fields that are available as parameters on the input tab.

Input

This tab displays the input schema of the activity in a tree structure format. The information in the schema varies based on the update query provided. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code the values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Substitution parameters from the query statement are provided in the Parameters node.

Output

The Update activity returns the number of rows affected by the query.

PostgreSQL Delete

Use this activity to execute an SQL Delete to delete the record and return the information based on the returning clause specified in the delete query.

Settings

The **Settings** tab has the following field:

Field	Description
Connection	Name of the PostgreSQL database connection from where information is retrieved. You can select the connection from the Connection drop-down list.

Input Settings

The **Input Settings** tab has the following fields:

Field	Description
Delete	<p>An SQL statement used to delete the record from the table. You can construct prepared SQL queries by using substitution variables (or parameters) of the form ?<fieldname> in the query statement.</p> <p>Each substitution variable identifies an input parameter whose mapped value is substituted into the substitution variable at runtime. Input and Output tabs are populated based on the query.</p> <p>End the query with semicolon (;) to fetch the metadata in the table at the bottom of the query.</p> <p>The following example represents a typical delete query:</p> <pre>Delete from student where name = ?name returning id;</pre> <p>For the above delete query, <code>output</code> field is generated for <code>id</code> from the returning clause and <code>input</code> field is generated for <code>name</code> (VARCHAR). In addition, the mapped value for the field name is substituted into the substitution variable ? name.</p>
Manually Configure Metadata	Set the Manually Configure Metadata field to <code>True</code> to fetch table metadata manually. For more information, see Manually Configure Metadata .
Fields	The grid is provided for informational purposes only.

Input

This tab displays the input schema of the activity as a tree structure. The information in the schema varies based on the delete query provided. The fields that are selected on the **Input Settings** tab are available in the schema. You can either hard code the values or map them to a field from the output of a preceding activity in the flow using the Mapper.

Output

The Delete activity returns the number of rows affected by the query.

Manually Configuring Metadata

You can use **Manually Configure Metadata** option to fetch table metadata manually.

When the field is set to `True`, one more **Query** field at the bottom of this option is displayed. To retrieve and configure the database schema fields, you must first enter a simple schema query in this field and click **Fetch**. For example, you can enter the following schema query to fetch the metadata:

```
SELECT * FROM CUSTOMERS;
```

After the successful processing of the query, the table field at the bottom is populated with the column metadata information. You can add, edit, and remove fields in the table. The fields marked **Selected** in the table are available on the **Output** tab of the activity. The fields selected in the **Parameter** column are available on the **Input** tab of the activity. The fields selected in the **Value** column are available on the **Output** tab of the activity. You can modify these settings by selecting or clearing the check boxes.

TIBCO Cloud Messaging

This section contains the following topics:

Overview

TIBCO Cloud™ Integration - Flogo® (PAYG) provides out-of-the-box connectivity for TIBCO Cloud Messaging and allows sending and receiving messages to and from TIBCO Cloud Messaging service.



Before you use the connector, ensure that you have a valid TIBCO Cloud Messaging account.



For details on using TIBCO Cloud Messaging, refer to the TIBCO Cloud Messaging product documentation.

Limitations

Keep the following limitations in mind when using the TIBCO Flogo® Connector for TIBCO Cloud™ Messaging:

- Boolean type and null data type are not supported in TIBCO Cloud Messaging.
- Anonymous arrays are not supported in TIBCO Cloud Messaging.
- TIBCO Cloud Messaging does not support multi-dimensional arrays.
- TIBCO Cloud Messaging does not support wildcards in content matcher.

Creating a TIBCO Cloud Messaging Connection

You must create a TIBCO Cloud Messaging connection before you can use the trigger or activity for sending and receiving messages to and from TIBCO Cloud Messaging service.. The TIBCO Cloud Messaging connection contains all the parameters required to connect to TIBCO Cloud Messaging. The TIBCO Cloud Messaging connection is used by all the activities in the TIBCO Cloud Messaging category.



Before you create a TIBCO Cloud Messaging connection, be sure that you have an active TIBCO Cloud Messaging subscription.



Before you create a connection, familiarize yourself with TIBCO Cloud Messaging. For details on how to use the TIBCO Cloud Messaging product, see the TIBCO Cloud Messaging product documentation.

To create a TIBCO Cloud Messaging connection, click the **Connections** tab on the TIBCO Cloud Integration - Flogo (PAYG) page.

If this is the first connection you are adding, do the following:

1. Click the **TIBCO Cloud Messaging Connector** tile.
2. Enter the values for the fields in the **TIBCO Cloud Messaging Connector** dialog. See [TIBCO Cloud Messaging Connection Details](#) topic for a description of the fields.
3. Click **Save**.

If you already have an existing connection for any connector in TIBCO Cloud Integration - Flogo (PAYG), the connections will be displayed on the **Connections** page.

1. Click **Add Connection**.
2. Click the **TIBCO Cloud Messaging Connector** tile.
3. Enter the values for the fields in the **TIBCO Cloud Messaging Connector** dialog. See [TIBCO Cloud Messaging Connection Details](#) topic for a description of the fields.
4. Click **Save**.

TIBCO Cloud Messaging Connection Details

Provide the information of the TIBCO Cloud Messaging connection to be created.

The **TIBCO™ Cloud Messaging Connection** dialog contains the following fields:

Field	Description
Connection Name	Name for the TIBCO Cloud Messaging connection that you are creating
Description	A short string of text describing the connection.
Connection URL	URL to the TIBCO Cloud Messaging service. You can obtain the URL from the TIBCO Cloud Messaging domain in the cloud.
Authentication Key	Authentication key required to log in to your TIBCO Cloud Messaging service. You can obtain the authentication key from the TIBCO Cloud Messaging domain in the cloud.
Timeout	Sometimes, a connection to the TCM server takes time. To avoid a connection timeout, specify a TCM Connection server timeout value in this field. Default: 10 seconds.
AutoReconnectAttempts	Specify the number of attempts that TIBCO Cloud Integration - Flogo (PAYG) should make to reconnect to the TCM server. By default, it is set to 25.
AutoReconnectMaxDelay	Specify the time in (seconds) you want TIBCO Cloud Integration - Flogo (PAYG) to wait between attempts to reconnect to the TCM server. The default value is 5 seconds.

MessageSubscriber Trigger

The Message Subscriber trigger listens for a published message from the TIBCO Cloud Messaging service.



Refer the [limitations](#) outlined in the [Overview](#) section before using this trigger.

Trigger Settings

The **Trigger Settings** tab has the following fields.

Field	Description
Connection	Name of the TIBCO Cloud Messaging connection.

Field	Description
Durable Subscriber	<p>By default, this field is set to False. Setting this field to True, changes the subscriber to a durable subscriber. As a durable subscriber, if the app goes down, any incoming messages during the period when the app was down will be preserved and delivered once the app comes back up again.</p> <p> A trigger marked as a durable subscriber should have been running at least once in order for it to receive pending messages that were sent while the app was down.</p> <p>When this field is set to True, the following fields appear:</p> <ul style="list-style-type: none"> • Durable Name: Enter a name for the subscription. This is a required field. • Durable Type: The following durable types are supported: <ul style="list-style-type: none"> – Shared- For multiple instances of an app, the messages received will be distributed among the app instances in a round-robin manner. – Standard-For every unique instance of an app, all the apps will receive all the messages. The messages will not be distributed. <ul style="list-style-type: none"> • In a single app, the same connection can not be used for two or more TCMMessageReceiver triggers and two or more handlers in a single trigger for Standard durable type. To configure Standard durable type for two more triggers or handlers create different connections. • In container deployment, if an app is scaled to more than one instance only the last instance receives all messages. • Message Ack Mode: The following acknowledgment types are supported: <ul style="list-style-type: none"> – Auto- In this mode, messages are auto acknowledged as soon as they are received from the TCM. In case of failure in business logic the messages are not redelivered. <p>Use this mode if message redelivery is not preferred.</p> – Explicit- In this mode, the message received by the MessageSubscriber Trigger must be explicitly acknowledged within the flow using TCMMessageAck activity. Any unacknowledged messages will be re-delivered to the new durable subscriber configured with the same durable name.
Destination	Name of the message destination. If this field is left blank then it will receive all messages, otherwise only the messages with the specific destination set.
Content Matcher	To specify the attributes that match the incoming message click Add row . For each attribute specify its name, type, and value.



You cannot have two subscribers with the same durable name but different destinations running concurrently. You must stop the subscriber that was already running before you start the second subscriber. This is also true for blue-green deployments supported by PaaS platforms, such as TIBCO Cloud Integration. So, do not update the destination name in blue-green deployments.

Output Settings

The fields selected in this tab will be available in the output schema in the **Map to flow Inputs** tab.

Field	Description
Message Schema	An example JSON object that you want to receive from TIBCO Cloud Messaging service.

Map to flow Inputs

This tab displays the schema specified in the **Output Settings** tab in a tree structure format. The output of an activity is displayed for informational purposes only and cannot be modified or altered.

The properties that are displayed in the **Output** tab schema correspond to the output of this activity and can be used as input by subsequent activities in the flow.

TCMMessagePublisher

This activity sends a message to the TIBCO Cloud Messaging service.



Refer the [limitations](#) outlined in the [Overview](#) section before using this trigger.

Settings

The **Settings** tab has the following fields.

Field	Description
Connection	Name of the TIBCO Cloud Messaging connection that you want to use.

Input Settings

The fields that you select in this tab will be available in the input schema in the **Input** tab.

Field	Description
Message Schema	A JSON object that you want to send to TIBCO Cloud Messaging.

Input

Field	Description
destination	The destination for a message.

In addition to **destination**, the **Input** tab displays the elements of the schema that you entered in the **Input Settings** tab in a tree format. You can input the values for each element by hard coding the value or mapping the value to an element from the output schema of a previous activity in the flow. See the section on Mapper in the TIBCO Cloud™ Integration - Flogo® (PAYG) documentation for details on how to map elements.

Loop

Refer to the section on "Using the Loop Feature in an Activity" in the *TIBCO Cloud™ Integration - Flogo® (PAYG) User's Guide* for information on the **Loop** tab.

Retry on Error

This tab allows you to set the number of times the flow should try to execute the activity if it encounters a retrievable error (such as waiting for a server to start, intermittent connection failures, or connection timeout) during the activity execution.

Map the elements in the schema using the mapper or alternatively, enter values for the element by manually typing the value in the mapper. See the section on "Using the Mapper" in *TIBCO Flogo® Enterprise User's Guide* for details on how to map elements.

Field	Description
Count	The number of times the flow should attempt to execute the activity.
Interval	The time (in milliseconds) to wait in between each attempt to execute the activity.

TCMMessageAck

The **TCMMessageAck** activity notifies the TCM Message Subscriber trigger to acknowledge the message received. This activity must be used when Message Ack Mode is set to **Explicit** on the trigger configuration.



The **TCMMessageAck** activity can be used in the main flow or error handler flow. It cannot be used in a subflow.

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, TIBCO Cloud, and Flogo are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see the readme.txt file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2016-2021. TIBCO Software Inc. All Rights Reserved.