

Using query tables in Business Process Choreographer Explorer

The iFix IZ66120 for WebSphere Process Server, version 6.2.0.2 enhances Business Process Choreographer Explorer to support query tables. Query tables enhance the predefined database views and the existing query interfaces of Business Process Choreographer. Query tables:

- Are optimized for running list queries, using performance optimized access patterns.
- Simplify and consolidate access to the information needed.
- Allow for the fine-grained configuration of authorization and filter options.

The iFix provides several default query table definitions for all of the entity types that are supported by Business Process Choreographer Explorer. These definitions are used for predefined views, personalized views, and adhoc searching. You can also define query tables for the specific access patterns of your business scenarios. In this way, you can provide various lists of tasks, processes, or activities, that contain the entities required for a particular usage scenario. You can combine data from multiple Business Process Choreographer views, and from external database tables.

If you already have query tables defined for your applications, they are also available to create custom views in Business Process Choreographer Explorer after you apply the iFix.

1. Software prerequisites

The iFix IZ66120 is applicable to Business Process Choreographer Explorer, version 6.2.0.2. The query table definitions that come with this iFix are built with SupportPac PA71: WebSphere Process Server - Query Table Builder, version 1.2. If you want to define and use your own query tables, use the latest version of PA71 available for Business Process Choreographer Explorer version 6.2.0.2.

2. Using query tables in Business Process Choreographer Explorer predefined views

The views that are initially available in Business Process Choreographer Explorer are all predefined views. After you install the iFix, the use of query tables for these views is transparent to users.

Most of the predefined views make use of query tables, except for the following views:

- *Process Templates* and *My Task Templates* – Query tables cannot take into account the state of the enterprise applications that contain the templates. Therefore, these views continue to be based on the EJB query API calls to ensure that they contain the expected content.
- *Failed Compensations* – This view lists microflows that had errors during compensation. There is no query table support for compensation.

- *Critical Processes* – This view lists all of the process instances that have at least one activity in the failed or stopped state. Currently, the query table API does not allow the specification of filter criteria that can return more than one matching entity.

3. Using query tables in Business Process Choreographer Explorer custom views

Custom views allow you to adapt Business Process Choreographer Explorer to your usage scenarios. The ability to define custom search criteria for a custom view based on the Business Process Choreographer query API is extended with this iFix to also allow using query tables. The default query tables that are delivered with this iFix are shown in the appendix of this document.

When you define a custom view, a list of available query tables for the entity is offered in a drop-down list on the search definition page. This list includes the default query table for the entity provided with the iFix, and any query tables that you have defined.


When using a default query table, you can change the columns to display, the sort order of the resulting items, the number of rows per page, and the threshold to be used. If you have SystemAdministrator or SystemMonitor rights, you can also limit the result to your personal items.


You cannot, however, define additional filter criteria on the default query tables. If you need additional filters, you can define them in the following ways:

- If you want to run an adhoc search, select **Do not use query tables** from the drop-down list on the search definition page and you can define filters.
- If you want to define a persistent custom view based on query tables, define your own query table with all the required filters using the Query Table Builder. Section 4 of this document provides best practices for defining a custom view in Business Process Choreographer Explorer based on a custom query table.

A typical business scenario needs specific business driven filters and columns - including query properties or custom properties. The sample query table definition provided with this TechDoc contains query properties. Also, a typical business scenario often needs variables in the query filter, for example, to search for a process with a specific customer number. You can use parameters in your query table definition to get this flexibility. This allows you to use a query table for more than one custom view in Business Process Choreographer Explorer. You can define default values for your parameters and specify whether the default values can be overwritten when the custom view is run. The sample query table definition provided with this TechDoc contains parameters.

When you modify a custom view based on a query table, you need to keep in mind the following information:

- After you save a custom view, you cannot change its query table. If you want to change the query table, create a copy of the view (by selecting the **Copy** icon ). Note that all of the filter, column, and sort settings are lost if you change the query table of a custom view.

- When you uninstall a query table in WebSphere Process Server, you also need to delete the related custom views in Business Process Choreographer Explorer (by selecting the **Delete** icon ).
- If you change a query table and deploy it again in WebSphere Process Server, you also need to redefine all of the related custom views in Business Process Choreographer Explorer.
- Do not modify or uninstall query tables that are provided by IBM. The names of these query tables are all prefixed with 'IBM.'

4. Best practices for defining query tables for Business Process Choreographer Explorer

This section contains best practices for using the Query Table Builder to define query tables for Business Process Choreographer Explorer. Section 5 of this document describes how to develop a sample query table based on these best practices.

4.1 General considerations

When you create a new query table definition, the Query Table Builder does not have an option for creating a query table for Business Process Choreographer Explorer. You should, therefore, select the option to create a **composite query table definition for Business Space**. This option ensures that all of the required columns are preselected so that the optimum performance gains can be achieved.

When you are finished developing your query table, test your query table definition in your business scenario to make sure that the query provides the expected results. Namely check, that you defined the correct authorization rules and filter criteria.

4.2 Authorization considerations

Consider the following authorization aspects when you define a query table.

- **Role-based authorization.**
Do not use role-based authorization for primary query tables that contain template information. Make sure that the related check box of the authorization property is unchecked when you define a primary query table based on templates. If you select the option to create a composite query table definition for Business Space, role-based authorization for templates is deselected by default. If you select role-based authorization for primary query tables that contain template information, you will see only those templates, for which you are authorized.
- **Instance-based authorization.**
Use the appropriate instance-based authorization filter for primary query tables that contain instance information. When you use query table support in Business Process Choreographer Explorer, this filter information must be set in the query table; you cannot set it in the corresponding User Roles tab in the custom view definition. Make sure that the related check boxes of the authorization property are set according to your business needs.

4.3 Internationalization considerations

Consider the following internationalization aspects when you define a query table.

- Display name and description for the query table definition
Provide a meaningful display name and description for all of the languages that are supported by your business.
- Display names and descriptions for columns
Business Process Choreographer Explorer retrieves the appropriate internationalized column names that are displayed in a result list. For columns that come from your primary query table (like PIID), you do not need to provide internationalized display names and descriptions. For these columns, Business Process Choreographer Explorer uses the translations that are already available for all the supported languages. For columns that come from an attached query table (like QUERY_PROPERTY), you need to provide meaningful display names and descriptions in the Query Table Builder in all of the languages that are supported by your business. So if you see a result list with cryptic column names in uppercase letters, you need to provide a matching translation for these columns.
- Task names and descriptions
If you have internationalized task names and descriptions in WebSphere Integration Developer, they are displayed in Business Process Choreographer Explorer according to the language and country settings of your browser. These settings must match the settings used when you defined the process model, otherwise the translation of the default language is used.
- Sort criteria
When you define sort criteria for a query table definition, be aware that several properties (for example, process state) are stored as integer values, while Business Process Choreographer Explorer displays them as translated strings in the resulting list. This might produce unexpected sorting results.

5. Samples for developing query tables

This section contains samples that you can use as step-by-step guides for building and using a custom query based on query table definitions. A related query table definition file is attached to this TechDoc for reference. The following samples are described here:

- Using query properties and parameters in a query table
- Globalizing your custom query

5.1 Using query properties and parameters in a query table

This sample shows how to define a query table, and create a personalized view based on this query table. The sample uses the ClaimProcess process model, which is available from the WebSphere Process Server Business Process Management Samples & Tutorials Web site. ClaimProcess demonstrates the usage of query properties. See the Related information section of this document for further information.

This sample is based on the scenario that the user frequently needs to look up the details of an insurance claim, which is identified by the customerID. In addition, the amount of the

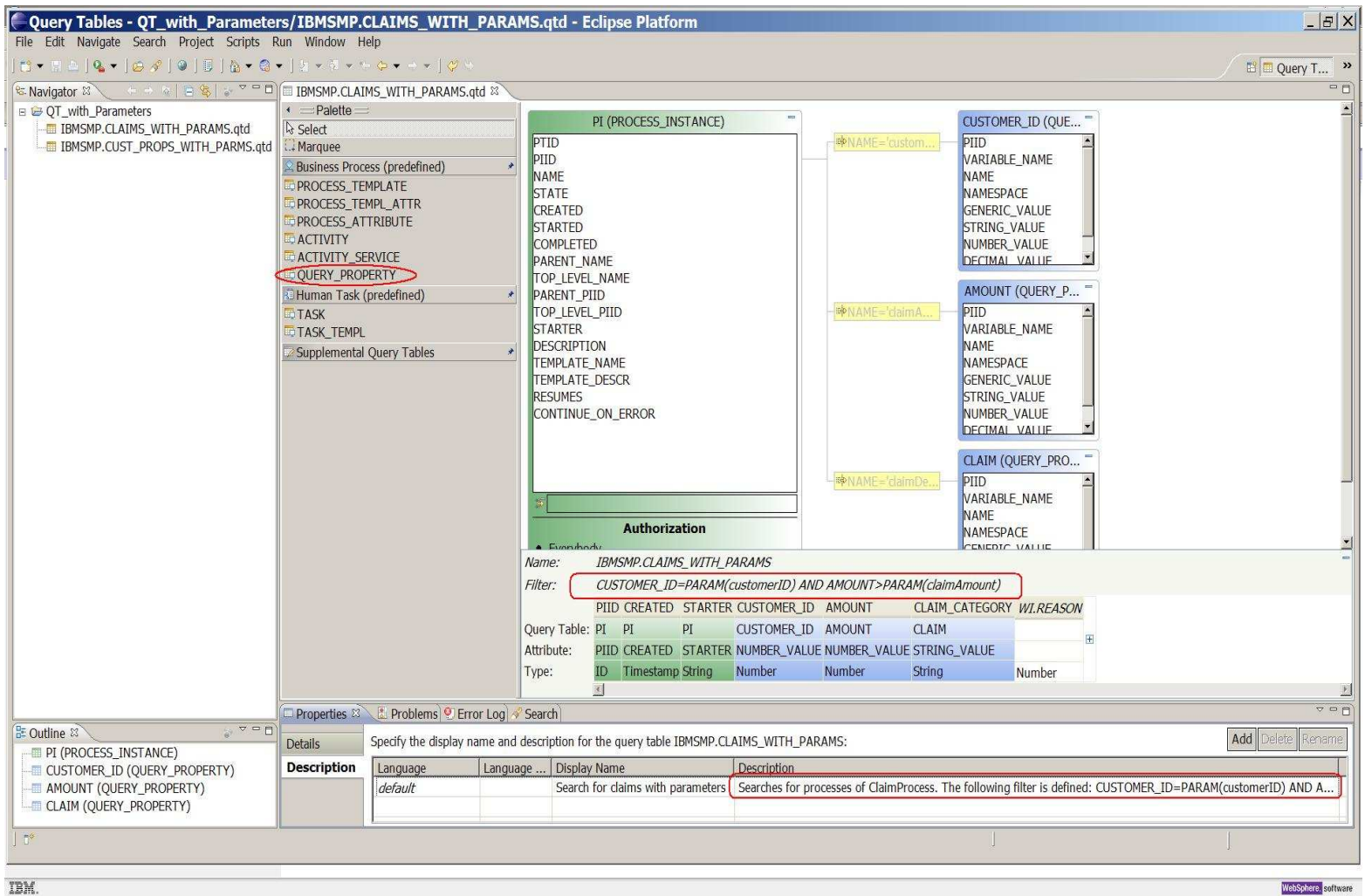
compensation request is used to limit the set of the qualifying claims. The user also needs to see the claim category, which describes the case.

This scenario requires a user-defined query table definition with parameters to support the custom view that is needed in Business Process Choreographer Explorer.

Step 1: Define the query table

In the Query Table Builder, complete the following steps:

1. Create a new Composite Query Table Definition and select `PROCESS_INSTANCE` as the primary query table.
2. Create the query properties.
From the palette, drag three `QUERY_PROPERTY` instances onto the canvas. Define the selection criteria for these query properties according to the query property definition in the ClaimProcess process model: 'customerID', 'claimAmount' and 'claimDescription'. You can also define a short name for the properties, or use the default values.
3. Define the columns in the database that you want to query.
 - Select `PIID`, `CREATED` and `STARTER` and drag them onto the query table definition.
 - From the customerID and claimAmount query property table, select the `NUMBER_VALUE` attribute and drag it onto the query table definition (customerID and claimAmount are of type 'Integer' in the process model).
 - From the claimDescription table, select `STRING_VALUE` and drag it onto the query table definition.
 - Select each of the added query property columns and define its properties in the Properties tab: define the column name, the display name and the description.
4. Define the filter and the description of the query table definition.
Click the canvas to get the query table properties.
 - On the Details tab, enter the query table description and the filter string.
`CUSTOMER_ID=PARAM(customerID) AND AMOUNT>PARAM(claimAmount)`
This defines the customerID and claimAmount attributes as query table parameters that must be provided when a query on this query table is processed.
 - On the Description tab, enter a display name and a description for the query table definition.
The display name is the name that is shown in Business Process Choreographer Explorer for the query table definition. The description is shown when the administrator selects the query table definition in the custom query. To assist the administrator when defining a parameterized custom query, you should show the administrator the filter definition in the description so that he knows what parameters to define in the custom query.



Step 2: Deploy the query table

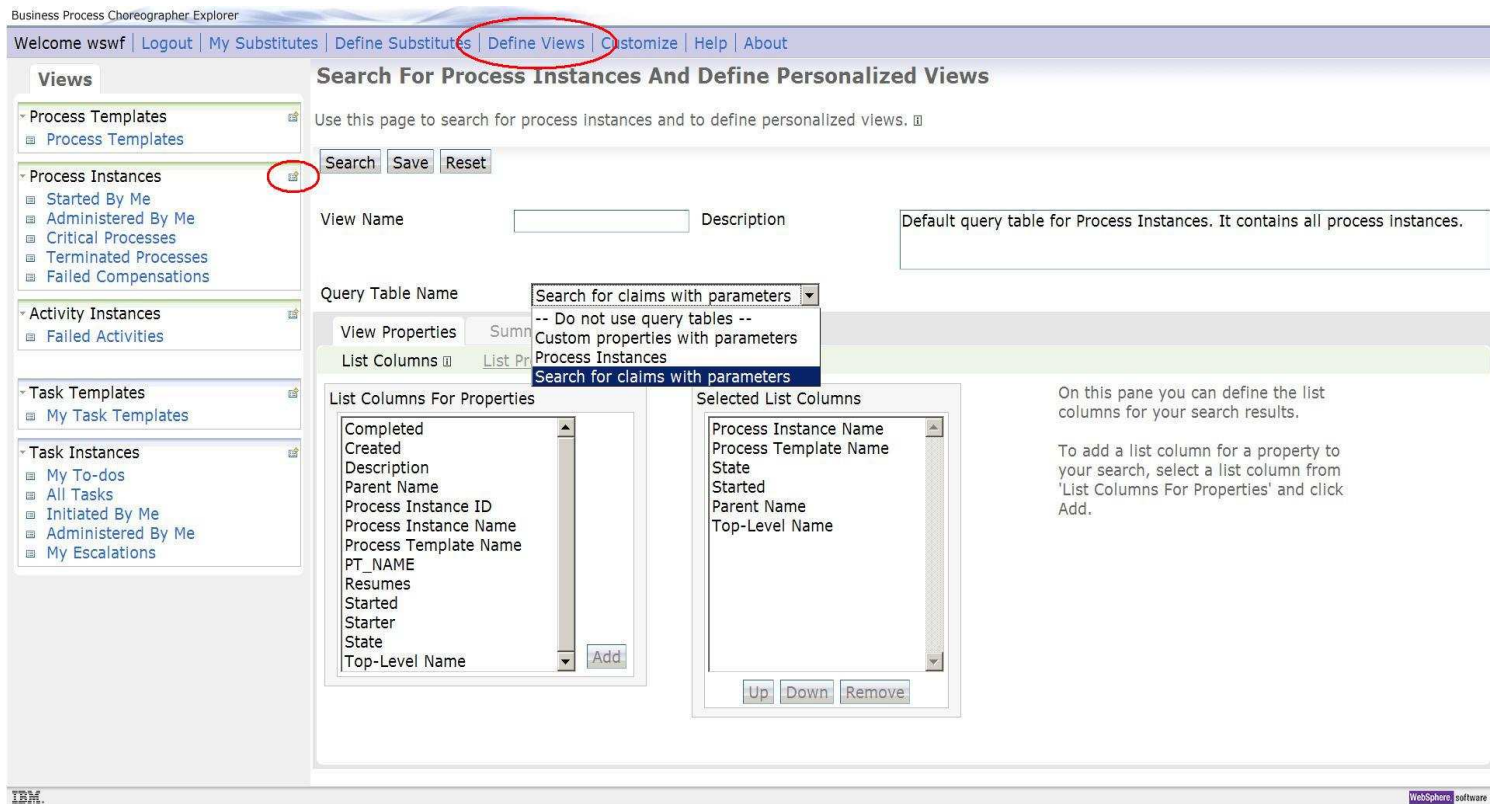
To make the query table definition available to an administrator, you need to export it from the Query Table Builder and deploy it to the WebSphere Process Server database.

1. Export the definition as a .jar file, including the .properties file so that the description and display name are contained.
2. Go to the `<install_root>/ProcessChoreographer/admin` directory, and use the `manageQueryTable.py` script to deploy the exported .jar file. For example, on Windows platforms, enter the following command :


```
..\..\bin\wsadmin -server <serverName>
-f manageQueryTable.py
-deploy <sampleQT.jar>
```

Step 3: Create a custom query based on the query table definition

As an administrator, log on to Business Process Choreographer Explorer. On the navigation pane in the Process Instances group, click the **New search** icon, or click **Define Views**. The Process Search page is displayed.



1. From the Query Table Name list, select your query table definition. You see the display name that you specified for it in the Query Table Builder.
2. Select the list columns for your custom query and the order in which they should be displayed.
3. In the Description field, you can add additional information for the user of your query to the existing description from the Query Table Builder. For example, you can add information about the expected query parameters and their value range. When a parameterized custom query is run, this description is displayed on the parameter input page where users specify the actual parameter values.
4. Specify the query parameters for the custom query on the Query Parameters tab. These parameters have to match the definitions of the query table, that is, the names and value ranges must match. Also, if applicable, add default values to help the user to understand the values that are expected.

Views

- Process Templates
 - Process Templates
- Process Instances
 - Started By Me
 - Administered By Me
 - Critical Processes
 - Terminated Processes
 - Failed Compensations
- Activity Instances
 - Failed Activities
- Task Templates
 - My Task Templates
- Task Instances
 - My To-dos
 - All Tasks
 - Initiated By Me
 - Administered By Me
 - My Escalations

Search For Process Instances And Define Personalized Views

Use this page to search for process instances and to define personalized views. ▢

View Name Description

Searches for processes of a specific customer. Enter the following search criteria: customerID (required) = the numeric ID of the claim requestor, claimAmount (optional) = the minimum estimated compensation amount

Query Table Name

Query Parameter Definition

Name	Default Value	
<input type="text" value="customerID"/>	<input type="text"/>	<input type="button" value="Remove"/>
<input type="text" value="claimAmount"/>	<input type="text" value="0"/>	<input type="button" value="Remove"/>
<input type="button" value="Add"/>		
<input type="checkbox"/> Do not prompt for the parameter values when using the view.		

On this pane you can specify the parameter and default values that are needed for the query table. You must specify all of the parameter names defined in the query table.

Tip: Provide default parameter values as help for the expected input format. Consider adding help information about parameter usage to the description field on this page of the view.

- Click **Search** to run the query. The parameter input page is displayed, where you can enter values for the parameters.

Views

- Process Templates
 - Process Templates
- Process Instances
 - Started By Me
 - Administered By Me
 - Critical Processes
 - Terminated Processes
 - Failed Compensations
- Activity Instances
 - Failed Activities
- Task Templates
 - My Task Templates
- Task Instances
 - My To-dos
 - All Tasks
 - Initiated By Me
 - Administered By Me
 - My Escalations

Specify query parameters for Search Customers

On this pane you can specify the parameter values for this run of the query.

Searches for processes of a specific customer. Enter the following search criteria: customerID (required) = numeric ID of the claim requestor, claimAmount (optional) = the minimum estimated compensation amount in \$

customerID
claimAmount

5.2 Globalizing your custom query

The scenario for this example is that your users are located in the United States, Germany, and France so you need to provide the query information in multiple languages. It extends the previous ClaimProcess example by adding support for German and French to the query table definition.

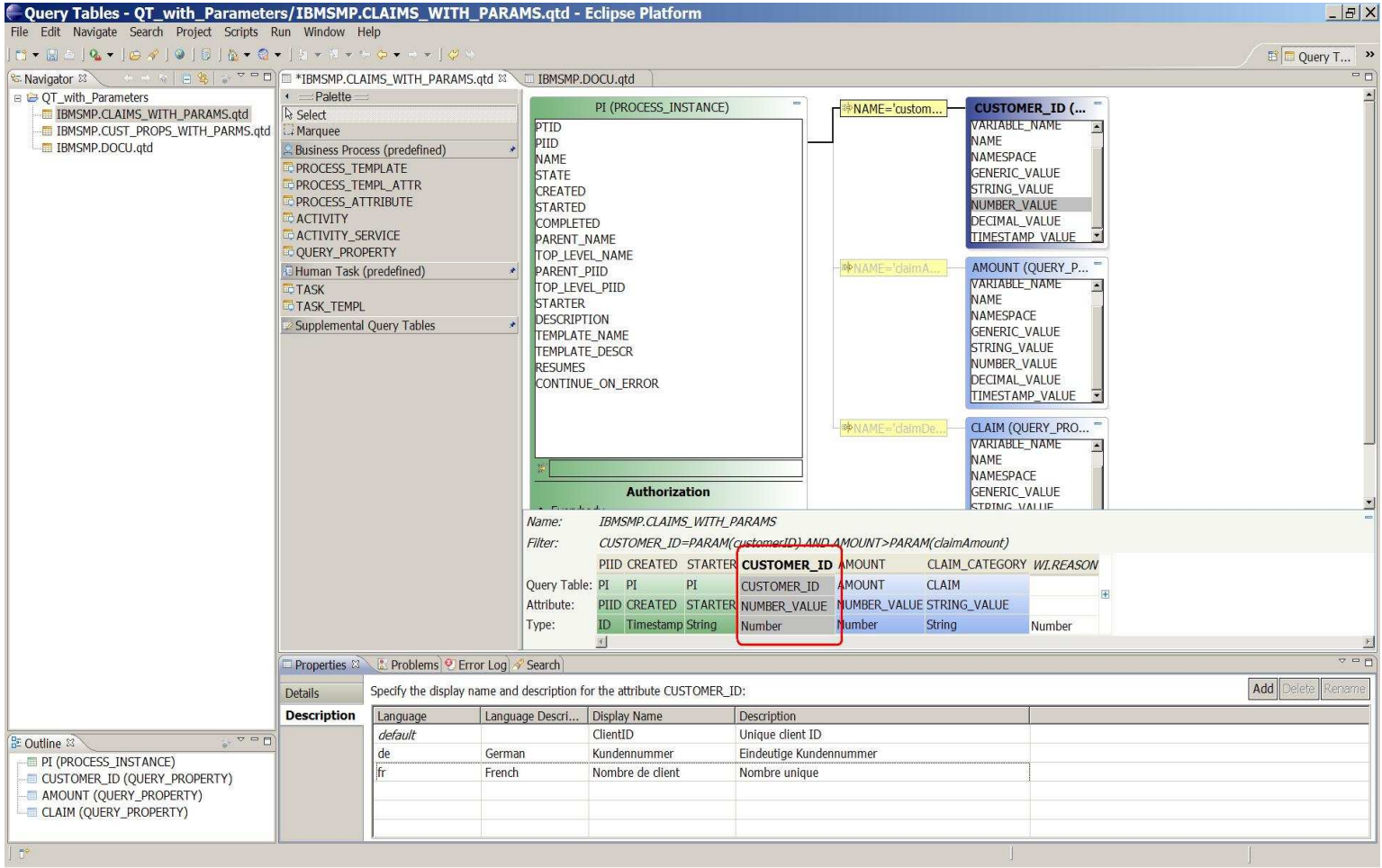
Step 1: Extend the query table

In the Query Table Builder, select the ClaimProcess query table definition, and open the Description tab of the query table properties.

The screenshot shows the Eclipse Platform Query Table Builder interface. The main window displays the query table definition for 'IBMSMP.CLAIMS_WITH_PARAMS.qtd'. A dialog box titled 'Add' is open, allowing the user to select additional languages. The 'Add' dialog shows a list of predefined languages: da_DK_EURO [Danish (Denmark,Euro)], de [German], de_AT [German (Austria)], de_AT_PREEURO [German (Austria,PREEURO)], and de_CH [German (Switzerland)]. The 'de' (German) option is selected. Below the list, there is a field to 'Specify a language manually:'. The main window also shows the 'Description' tab of the query table properties, which contains a table with columns for Language, Language Description, Display Name, and Description. The 'Add' button in the 'Description' tab is highlighted with a red box.

Language	Language Description	Display Name	Description
default		Search for claims with parameters	Searches for processes of ClaimProcess. The following filter is defined: CUSTOMER_ID=PARAM(customerID)

1. To add additional languages, click **Add**, and select de (German), and fr (French).
2. Insert a display name and description for the query table in German and French. This information will be available when an administrator defines a custom query. Depending on the browser's language setting, the translated query table name and description is displayed. If the browser's language setting is de-XX or fr-XX the German or French query table name is provided, otherwise the name of the default language is used (English in this sample).
3. Provide translations for the other columns that you want to display. Select each of the query properties columns defined for your custom query and provide German and French translations for the display name and description. Note, you do not need to provide translations for the standard columns of the process instance as this is already translated by Business Process Choreographer Explorer.



Step 2: Deploy the query table

To make the updated query table definition available to an administrator, you need to export it from the Query Table Builder and deploy it to the WebSphere Process Server database.

1. Export the definition as a .jar file, including the .properties file so that the description and display name are contained.
2. Go to the `<install_root>/ProcessChoreographer/admin` directory, and use the `manageQueryTable.py` script to deploy the exported .jar file. For example, on Windows platforms, enter the following command :

```

..\..\bin\wsadmin -server <serverName>
                  -f manageQueryTable.py
                  -update definition <sampleQT.jar>

```

Step 3: Use the globalized query table in Business Process Choreographer Explorer

As an administrator, log on to Business Process Choreographer Explorer. On the navigation pane in the Process Instances group, click **Define Views**. The Process Search page is displayed. The query table name is translated according to your browser's language settings. Define custom queries or run adhoc queries using this query table definition. When the

result is displayed, you can see that the query properties columns are translated according to your browser's language settings.

Business Process Choreographer Explorer

Willkommen wswf | Abmelden | Meine Vertreter | Vertreter definieren | Sichten definieren | Anpassen | Hilfe | Produktinfo

Sichten

- Prozessschablonen
 - Prozessschablonen
- Prozessinstanzen
 - Von mir gestartet
 - Von mir verwaltet
 - Kritische Prozesse
 - Beendete Prozesse
 - Fehlgeschlagene Kompensationen
- Search Customers
- Aktivitätsinstanzen
 - Fehlgeschlagene Aktivitäten
- Taskschablonen
 - Meine Taskschablonen
- Taskinstanzen
 - Meine unerledigten Tasks
 - Alle Tasks
 - Von mir eingeleitet
 - Von mir verwaltet
 - Meine Eskalationen

Suchergebnisse für Prozessinstanzen

Auf dieser Seite können Sie mit den Ergebnissen Ihrer Suche nach Prozessinstanzen arbeiten.

Status des Prozesses anzeigen | Zugehörige Prozesse | Aktivitäten | Tasks | Aktualisieren | Sicht bearbeiten

<input type="checkbox"/> ID der Prozessinstanz	Erstellt	Starter	Kundennummer	Schadenssumme	Schadensart
<input type="checkbox"/> _PI:90030125.b1dd3adc.96896df6.d5bb01bb	21.12.2009 16:31	wswf	4567	100000	House damaged

Gefundene Elemente: 1 Ausgewählte Elemente: 0 Seite 1 von 1 Elemente pro Seite: 20

Appendix: Default query tables

A default query table is available for each entity that is supported in Business Process Choreographer Explorer. This query table mimics the existing behavior of running a search directly without additional filter criteria. The following definitions are used:

Entity / Query Table Name	Columns (*preselected as visible)	Filter
Process Templates	Process Template Name* Valid From* Long Running* State* Description* Technical Name Process Template ID Namespace Application Name Created Version	n/a
Process Instances	Created Description Parent Name* Process Instance ID Process Instance Name* Process Template Name* Technical Template Name Resumes Started* Starter State* Top-Level Name*	Checks authorization for individuals, groups and 'everybody'.
Activity Instances	Activated* Activity Instance ID Activity Name* Completed Description Expires on Kind* Owner* Process Instance Name* Process Template Name* Technical Process Template Name Skip requested* Started State*	Checks authorization for individuals, groups and 'everybody' on activities, and the inherited authorization from the corresponding process instance. Filters for: (Kind in ('KIND_ASSIGN, KIND_COMPENSATE, KIND_CUSTOM, KIND_EMPTY, KIND_INVOKE, KIND_PICK,

		KIND_RECEIVE, KIND_REPLY, KIND_RETHROW, KIND_SCRIPT, KIND_STAFF, KIND_TERMINATE, KIND_THROW, KIND_WAIT') OR STATE = 'stopped')
Task Templates	Business Category* Description* Kind* Namespace* State* Task Template Name* Technical Task Template Name	Retrieves the task template name for the default locale used while modeling the task templates.
Task Instances	Activated* Ad Hoc Automatic Claim Business Category Business Relevant Claim if Suspended Completed Containment Context ID Delegation Description* Due Escalated* Expires* First Activated Inline Kind* Last Modified* Last State Change Namespace Originator* Owner* Parent Context ID Position in Hierarchy Priority* Resumes Started Starter State*	Checks authorization for individuals, groups and 'everybody'. Filters for: (KIND IN (KIND_HUMAN, KIND_ORIGINATING, KIND_PARTICIPATING, KIND_WPC_STAFF_ACTIVITY)) Retrieves descriptions for the default locale used while modeling tasks.

	<p>Subtasks Suspended* Task ID Task Name* Task Template Name Technical Task Name Technical Task Template Name</p>	
--	---	--

Related information

1. Information Center: Query Tables for Business Process Choreographer
http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r2mx/index.jsp?topic=/com.ibm.websphere.bpc.620.doc/doc/bpc/c6bpel_query_busproctask.html
2. SupportPac PA71: WebSphere Process Server - Query Table Builder
<http://www-01.ibm.com/support/docview.wss?rs=693&context=SSBTEG&q1=utility&uid=swg24021440>
3. ClaimProcess sample for interaction with processes and human tasks on the Business Process Management Samples & Tutorials page
<http://publib.boulder.ibm.com/bpcsamp/processInteraction/queryProperties.html>