



Xtraction Data Model Editor

User Guide

Table of Contents

Introduction	4
Starting the Data Model Editor	4
Editing a data model	7
Creating a new data model	7
Creating a new data source	7
Creating/editing a view	8
Creating/editing a table	13
Examples of complex fields	28
Binary Yes/No field	28
Time field in hours	29
Time field in periods	30
Concatenation of String fields	32
Data model Find & Replace	32
Additional Find options	33
Searching inactive references	35
Connection String Editor	36
URL Editor	37
Schema/Owner Editor	38
Copying objects	38
Saving a data model	40
Data model settings	40
Data model Schema View	41
Example data model	42

This document is provided strictly as a guide. No guarantees can be provided or expected. This document contains the confidential information and/or proprietary property of Ivanti, Inc. and its affiliates (referred to collectively as “Ivanti”) and may not be disclosed or copied without prior written consent of Ivanti.

Ivanti retains the right to make changes to this document or related product specifications and descriptions, at any time, without notice. Ivanti makes no warranty for the use of this document and assumes no responsibility for any errors that can appear in the document, nor does it make a commitment to update the information contained herein. For the most current product information, please visit www.ivanti.com.

Copyright © 2023, Ivanti, Inc. All rights reserved.
Rev 02/23

Introduction

This guide provides details for how to use the Xtraction Data Model Editor tool to configure a data model for use with Xtraction. Xtraction uses the data model file, named **DataModel.dat**, to connect to the various data sources it reports on.

A data model is made of the following objects:

- **Data source:** An application database that Xtraction is connected to.
- **View:** A segment of the data within a data source.
- **Table:** A table contained within a view. A table can be a physical data source object (i.e., a database table or database view) or if the DBMS allows, a virtual object (i.e., SQL statement).
- **Field:** A field contained within a table. A field can also be a physical or virtual object (i.e., SQL expression).

Starting the Data Model Editor

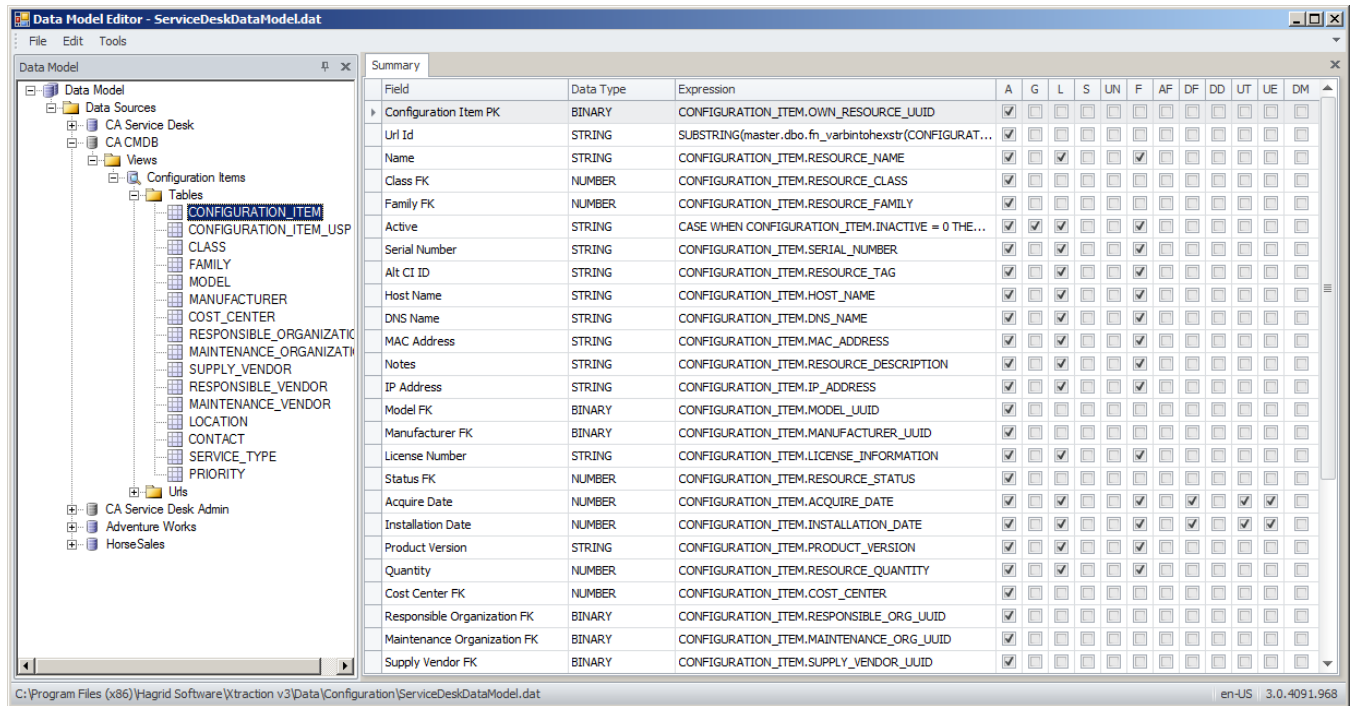
During the Xtraction installation process, the Data Model Editor files (.exe and .lic) are placed in the ...\\Program Files\\Xtraction Software\\Xtraction\\Tools folder on your Xtraction application server.

The Data Model Editor is licensed for use on a specific computer based on the value of the HOSTNAME environment variable. The tool will fail to start if a valid **DataModelEditor.lic** file is not installed and located in the ...\\Tools folder mentioned above.

NOTE: Most users remotely connect to the server and run the DataModelEditor.exe from the server itself. For details about running the Data Model Editor from a different computer, see this knowledge article [How do I install the DME on my workstation?](#)

Start the Data Model Editor by running the **DataModelEditor.exe** file. The file will take a moment to launch as it loads controls and libraries – so be patient.

The Data Model Editor initially opens with no data model loaded and displays an empty window. The sample window shown below has a data model loaded for instructional purposes. There are four separate areas within the main window:

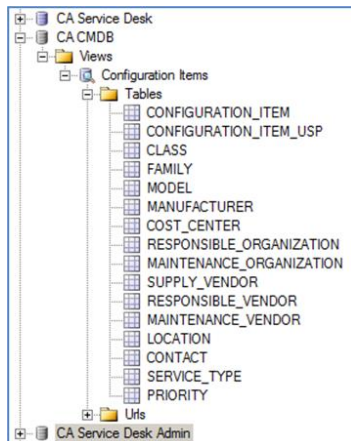


Menu bar: Although the menu options are reasonably self-explanatory, they will be covered further in the examples included throughout this guide.

Status bar: Located at the bottom of the main window, this bar displays the path and name of the current data model file, as well as the current language and version.

Data source pane: The left pane lists the data sources added to the open data model file. You can expand the tree view to see the views contained within the data sources, and tables contained within the views. Tables and URLs are grouped in folders.

Double-click a listed data source, view, or table to open a dialog for that object. When you select a table in the data source pane, the fields associated with that table are listed in the field pane on the right.



To add a new data source, right-click in the background or right-click a data model label to access a context menu. To add a new view, right-click a data source label. Right-click context menus exist for the other tree items as well and are explained later in this guide.

Field pane: The right pane displays a grid view of the fields contained within the table currently selected in the data source pane. The table is identified in the header, which is in the format **data source \ view \ table**. The table name is displayed using the text attribute, followed by the table attribute in parentheses.

Summary

Fields - CA Service Desk \ Incidents \ REQUEST (DBO.CALL_REQ)														
Field	Data Type	Expression	A	G	L	S	UN	F	AF	DF	DD	UT	UE	DM
Request PK	NUMBER	REQUEST.ID	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Type	STRING	REQUEST.TYPE	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
PersId	STRING	REQUEST.PERSID	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Number	STRING	REQUEST.REF_NUM	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Summary	STRING	REQUEST.SUMMARY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Status FK	STRING	REQUEST.STATUS	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Active	STRING	CASE REQUEST.ACTIVE_FLAG WHEN 1 THEN 'Yes' ELSE...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Open Date	NUMBER	REQUEST.OPEN_DATE	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

The attributes of each field in the table are displayed in a grid and are explained in detail later in the guide:

A = Active	S = Summary	AF = Admin filter	UT = UTC date
G = Group	UN = Unicode	DF = Date filter	UE = UNIX epoch
L = List	F = Filter	DD = Default date	DM = Data model

Editing a data model

You can use the Data Model Editor to create and modify the data model needed for your environment. Because the data model file (DataModel.dat) enables the connection between Xtraction and a specific type of data repository, it's often referred to as a **connector**.

Ivanti provides two types of pre-built connectors whose settings you can edit in the Data Model Editor after downloading:

- Connectors for Ivanti's own products, provided free of charge and available from the [Connector Downloads](#) site (where you must first register to log in).
- Connectors for third-party products, licensed separately and available from the [Ivanti License Portal](#) (where you must sign in with the credentials provided by Ivanti).

For a list of available connectors (last updated January 2021), see the PDF [Ivanti Xtraction Connectors](#).

For details about the process of downloading the connectors, see the knowledge article [How to download Xtraction licenses and connectors from Ivanti](#).

When constructing a new data model file, you can either start from an empty file or copy an existing file to modify. In either case, you can copy objects between or within data model files.

Before editing a data model file, always make a backup copy in case you need to reload the previous model to revert your changes. Save files in the ...\\Program Files\\Xtraction Software\\Xtraction\\Data\\Configuration\\ folder.

NOTE: In the following sections, the options for File > New, Add Data Source, and Add View are only available if you have purchased an Enterprise Data Model Editor license.

Creating a new data model

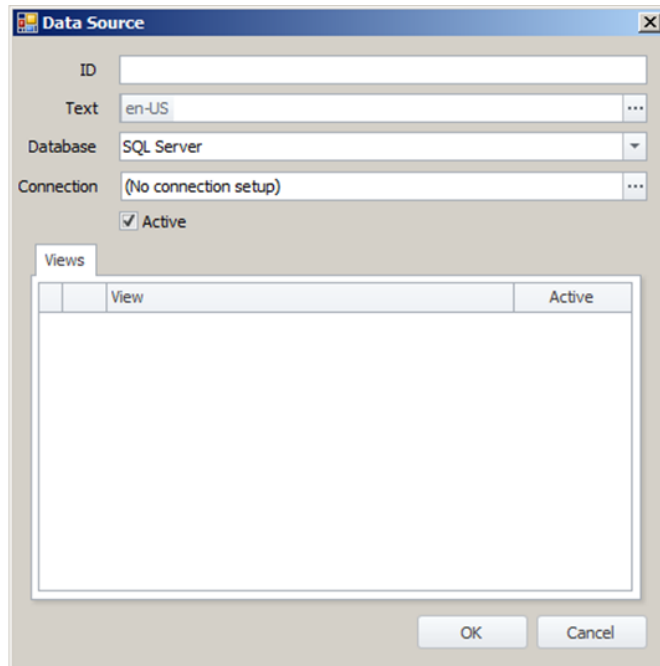
The Data Model Editor starts without a pre-configured data model loaded, which enables you to build and save a new data model file from scratch.

When starting from an empty data model, you'll need to create the data sources, views, tables, fields, etc., as described in the following sections.

At any time, you can discard the current data model and revert to an empty data model by clicking the **File > New** menu.

Creating a new data source

Right-click anywhere in the data source pane and select **Add Data Source**. The Data Source dialog displays, enabling you to enter various details pertaining to the data source, as required by Xtraction.



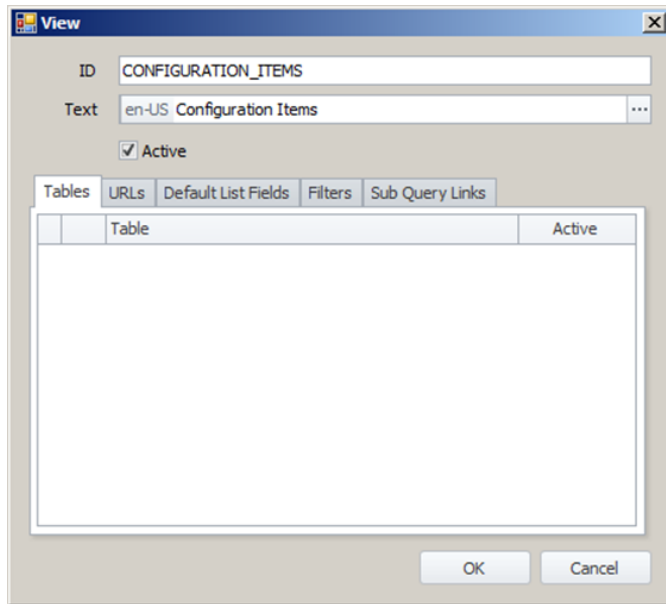
Fill in the following fields:

- **ID:** A text field used to identify the data source internally. Enter text strings in uppercase letters (and optionally numbers) with no other characters except underscores, which should be used for spacing (e.g., CA_CMDB).
- **Text:** A text field used to identify the data source in the user interface (i.e., the Data Model Editor and Xtraction). These text strings do not have the same restrictions as the ID attribute and may contain mixed case letters, spaces, and special characters.
- **Database:** From the drop-down list, select the relevant database management system for the data source.
- **Connection:** Specify the details necessary for Xtraction to connect to the database. Click the ellipsis button to open the Connection Details dialog, where you can specify the server and database names, user ID and password, etc. Additional information is optional. Click **Test** to initiate a connection to the database. If successful, you'll see a box pop up that confirms the connection details work.
- **Active:** A binary field that enables you to work on new data sources and keep them inactive until they're complete and ready to integrate with Xtraction. When the data source is ready to integrate, click the **Active** option.

You can add views to the data source as detailed in the following section.

Creating/editing a view

To create a new view for a data source, double-click a data source name in the data source pane. From the open Data Source dialog, right-click in the **Views** tab area and click **Add**. The View dialog displays, as shown below.



Fill in the following fields:

- **ID:** A text field used to identify the view internally. Enter text strings in uppercase letters (and optionally numbers) with no other characters except underscores, which should be used for spacing (e.g., CONFIGURATION_ITEMS).
- **Text:** A text field used to identify the view in the user interface (i.e., the Data Model Editor and Xtraction). These text strings do not have the same restrictions as the ID attribute and may contain mixed case letters, spaces, and special characters.
- **Active:** A binary field that enables you to work on new views and keep them inactive until they're complete and ready to integrate with Xtraction. When the view is ready to integrate, click the **Active** check box.

The View dialog also includes tabs for customizing the tables, URLs, default list fields, sub query links, and filters defined for the view.

View dialog > Tables tab

Right-click in the Table tab area to add, edit, or delete tables associated with the view. Details about adding and editing tables is covered in a section below.

When multiple tables have been added to a view, you can select a table in the list and right-click to move the table up or down the list. When you have many tables, it's good practice to reposition them in the list to maintain a rational order.

View dialog > URLs tab

Right-click in the URLs tab area to specify the URLs associated with the view. The URLs can enable Xtraction users to launch other resources (e.g., application windows) in context by double-clicking entries in an Xtraction record list.

The screenshot shows a dialog box titled "URL". It contains the following fields and controls:

- ID:** A text field containing "CA_SERVICE_DESK_RECORD".
- Text:** A text field containing "en-US CA Service Desk Record".
- URL:** A text field containing "http://win2008_r2/CAisd/pdmweb.exe?OP=SHOW_DETAIL+FACTORY=nr+PE".
- Active:** A checked checkbox.
- Fields:** A section containing two drop-down menus and an "Add" button.
 - The first drop-down menu is labeled "CONFIGURATION_ITEM".
 - The second drop-down menu is labeled "Configuration Item PK".
 - The "Add" button is to the right of the second drop-down.
- Table:** A table with two columns: "CONFIGURATION_ITEM" and "Url Id". It is currently empty.
- Buttons:** "OK" and "Close" buttons at the bottom right.

Fill in the following fields:

- **ID:** A text field used to identify the URL internally. Enter text strings in uppercase letters (and optionally numbers) with no other characters except underscores, which should be used for spacing (e.g., CA_SERVICE_DESK_RECORD).
- **Text:** A text field used to identify the URL in the user interface (i.e., the Data Model Editor and Xtraction). These text strings do not have the same restrictions as the ID attribute and may contain mixed case letters, spaces, and special characters.
- **URL:** A text field used to specify the full URL. The URL can refer to one or more fields from the data returned. You can add fields from the data model by selecting them from the drop-down lists and clicking the **Add** button.

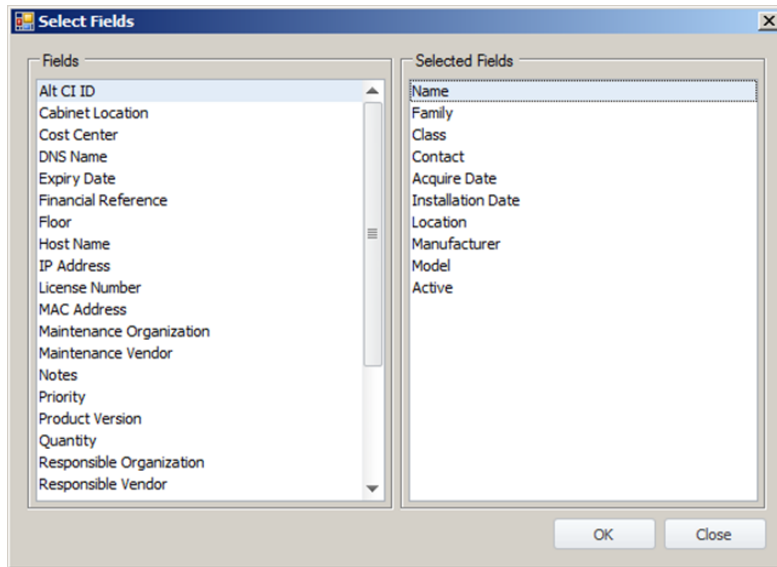
To reference a field within the URL string, you need to refer to a field's position, beginning at 0. In the example above, there is one field set up, **Url Id**, and it's being referenced in the URL string via {0}: `http://win2008_r2/CAisd/pdmweb.exe?OP=SHOW_DETAIL+FACTORY=nr+PERSID=nr:{0}`

If a second field had been set up, it could be referenced in the URL string as {1}.

- **Active:** A binary field that enables you to work on new URLs and keep them inactive until they're complete and ready to integrate with Xtraction. When a URL is ready to integrate, click the **Active** check box.

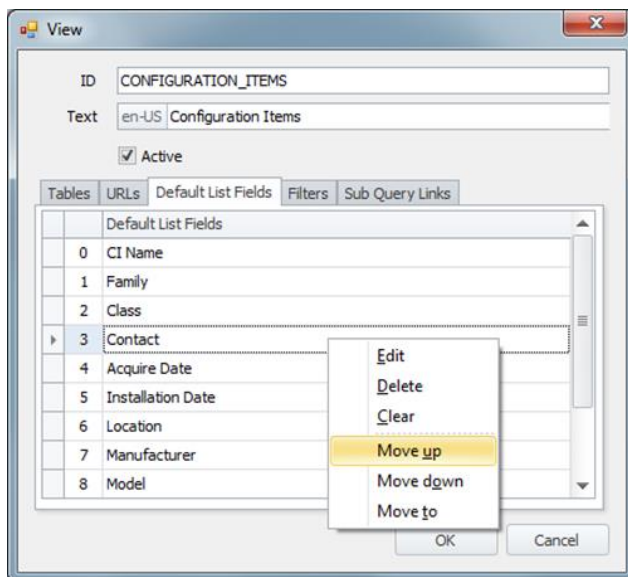
View dialog > Default List Fields tab

Specify the default fields and the order in which they're presented when records from the view are displayed within lists in Xtraction. To modify the list fields, right-click in the Default List Fields tab area and select **Edit**. The Select Fields dialog displays, allowing you to select and order the default list fields.



You can select any of the available fields in the left column and drag them over to the right column.

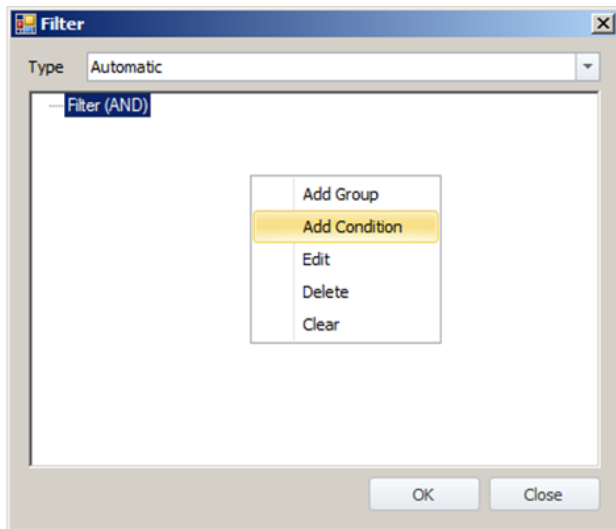
You can't reorder or delete fields in this dialog, but you can do so back on the Default List Fields tab using the right-click menu as shown below.



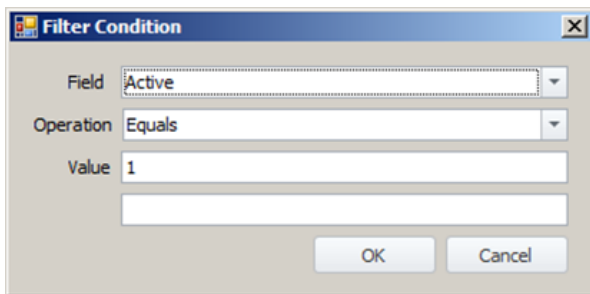
View dialog > Filters tab

You can specify filters that should be applied to the view when Xtraction retrieves record data. To modify the filters, right-click in the Filters tab area and select from options to add, edit, or delete.

The Filter dialog will display if you're adding or editing a filter, as shown in the image below. You can then right-click in the Conditions pane and select **Add Condition**.



An example of the Filter Condition dialog is shown below.

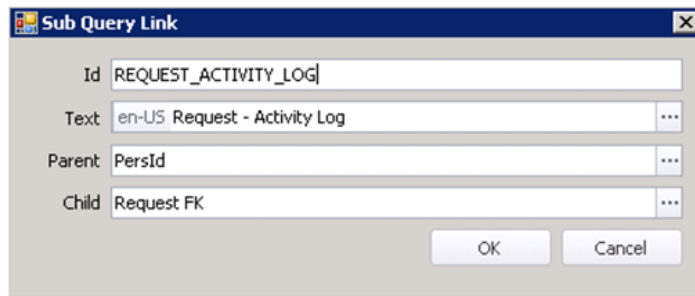


Set the **Field** and **Operation** from the drop-down lists. Set the **Value** according to the field values you want to filter in the record data returned by Xtraction.

View dialog > Sub Query Links tab

Sub query links are used as the basis for sub query filter conditions within Xtraction (for details, see the online User Guide). Think of a sub query filter condition as a filter within a filter, and for Xtraction to join the main filter with the sub query filter condition, a link between fields is required. Sub query links are mainly used in situations where there is a one-to-many type relationship. They allow filter conditions to be set up on the many side without returning multiple records.

To add a sub query link, right-click in the Sub Query Links tab area and select **Add**. To edit an existing sub query link, double-click the link text.



The 'Sub Query Link' dialog box contains the following fields:

- Id:** REQUEST_ACTIVITY_LOG
- Text:** en-US Request - Activity Log
- Parent:** PersId
- Child:** Request FK

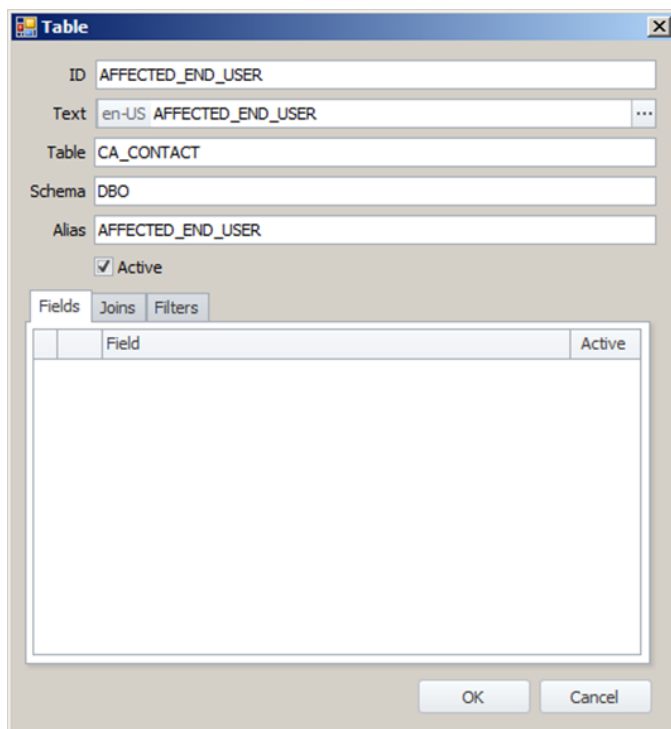
Buttons: OK, Cancel

Like other objects within the data model, you need to provide an ID and text. The text you enter will be displayed within Xtraction when a user sets up a sub query filter condition. Lastly, select the two fields that are to be linked together. Click **OK** to create the link.

Creating/editing a table

To create a new table for a view, double-click the data source view name in the data source pane.

From the View dialog, right-click the **Tables** tab and select **Add**. The Table dialog will display, enabling you to enter various details pertaining to the table, as required by Xtraction.



The 'Table' dialog box contains the following fields:

- ID:** AFFECTED_END_USER
- Text:** en-US AFFECTED_END_USER
- Table:** CA_CONTACT
- Schema:** DBO
- Alias:** AFFECTED_END_USER
- ☒ **Active**

Buttons: Fields, Joins, Filters, OK, Cancel

Field	Active

Fill in the following fields:

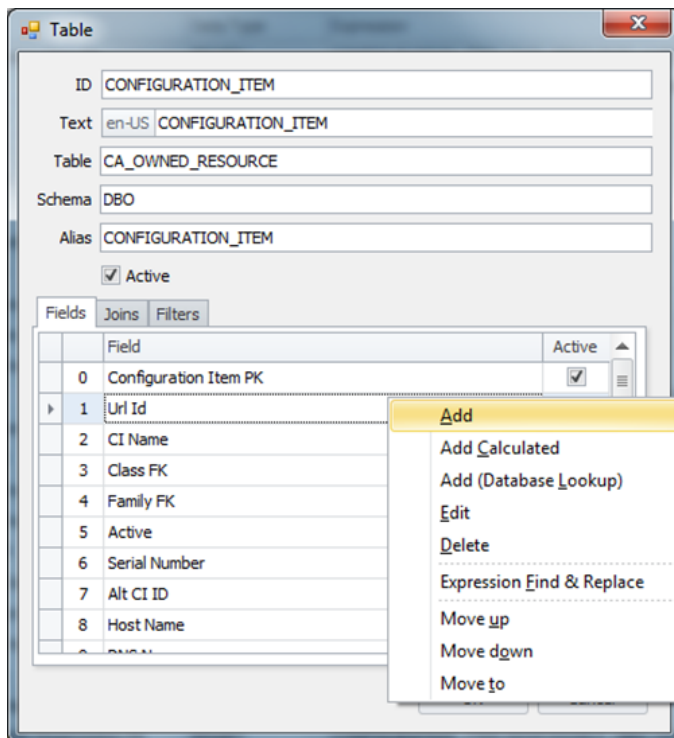
- **ID:** A text field used to identify the table internally. Enter text strings in uppercase letters (and optionally numbers) with no other characters except underscores, which should be used for spacing (e.g., AFFECTED_END_USER).

- **Text:** A text field used to identify the table in the user interface (i.e., the Data Model Editor and Xtraction). These text strings do not have the same restrictions as the ID attribute and may contain mixed case letters, spaces, and special characters.
- **Table:** A text field used to select the applicable table within the database.
- **Schema:** A text field used to select the applicable schema within the database.
- **Alias:** A text field used to provide an alternate query alias for the table.
- **Active:** A binary field that enables you to work on new tables and keep them inactive until they're complete and ready to integrate with Xtraction. When a table is ready to integrate, click the **Active** check box.

Table dialog > Fields tab

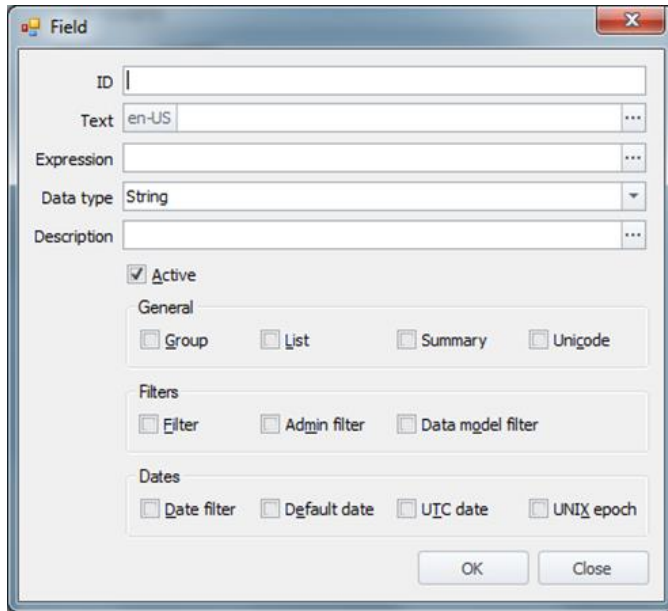
The Fields tab enables you to add, edit, or delete fields by using a right-click menu. When multiple fields have been added to a table, you can use the same menu to move fields up or down the list. When you have many fields, it's good practice to reposition them in the list to maintain a rational order.

The **Expression Find & Replace** option on the right-click menu enables you to locate and replace text strings. It's explained in a section below.



The **Add** option is useful when you're designing a field that's not in the database or when you're currently unable to connect to the database. The attributes are explained in a section below.

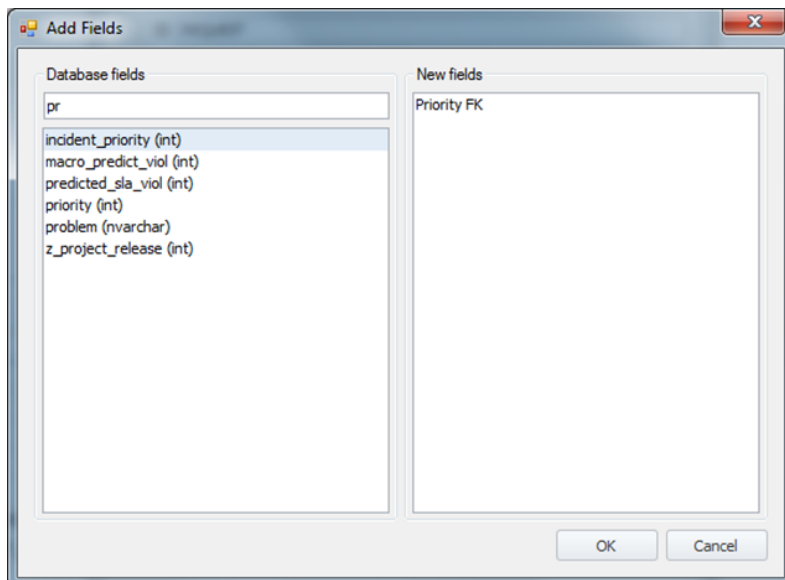
When adding a field, only two of the attributes are preset (you may need to change the Data type) and one attribute is partially preset (you'll need to complete the Expression).



When you can connect to the database, the **Add (Database Lookup)** option can interrogate the database for available fields and preset more attributes for your selected field.

When selecting the Add (Database Lookup) option, an Add Fields dialog displays, as shown in the image below. Database fields are displayed in the left pane, and you can drag one field at a time into the New Fields pane on the right.

Note that you can filter the database field list for fields containing a specific value (below, the list is filtered to display fields containing “pr” in their name).



As you drag each field into the right pane, a Field dialog opens with preset attributes for the selected database field, as shown in the next image.

Fill in the following fields:

- **ID:** A text field used to identify the field internally. Enter text strings in uppercase letters (and optionally numbers) with no other characters except underscores, which should be used for spacing (e.g., PRIORITY_FK).
- **Text:** A text field used to identify the field in the user interface (i.e., the Data Model Editor and Xtraction). These text strings do not have the same restrictions as the ID attribute and may contain mixed case letters, spaces, and special characters.
- **Expression:** A text field used to specify the SQL syntax required to return the desired value. Click the ellipse button to enlarge the edit box into a multi-line edit window. The simplest form of these expressions is written in the TABLE.FIELD format, as shown above (i.e., REQUEST.PRIORITY). Examples of more complex expressions are detailed in the Examples of Complex Fields section of this guide.
- **Data type:** Select from a drop-down list with options including String, Number, Date, Boolean, and Binary. An Other option is for unlisted data types, which may require non-standard expressions.
- **Description:** A text field used to display a description or tooltip in Xtraction when the mouse hovers over the column in various places including a List Component column. Click the ellipse button to enlarge the edit box into a multi-line edit window.

The remaining attributes are binary fields set by either checking or clearing a check box:

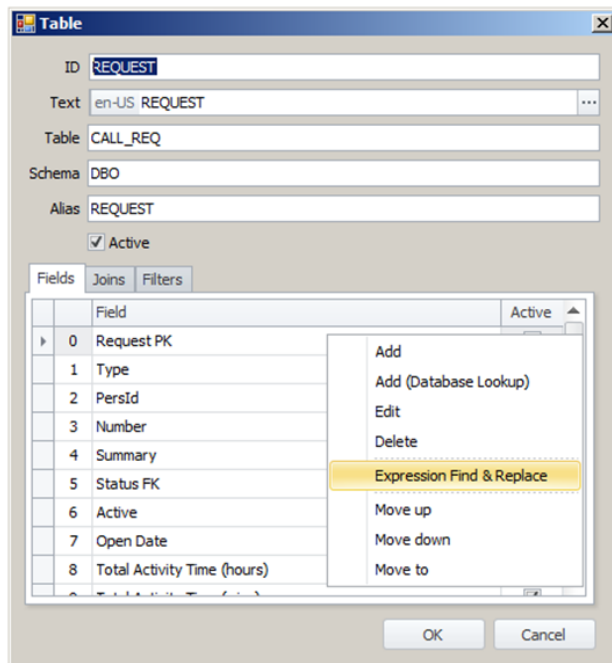
- **Active:** Enables you to work on new fields and keep them inactive until they're complete and ready to integrate with Xtraction.
- **Group:** Determines whether the field is included as a Group field in Xtraction.
- **List:** Determines whether the field is included as a List field in Xtraction.
- **Summary:** Determines whether the field is included within the Summary Options in Xtraction.
- **Unicode:** Determines whether the field is treated as a Unicode field in Xtraction.
- **Filter:** Determines whether the field can be included within Filter Conditions in Xtraction.

- **Admin filter:** Determines whether the field can be included within Admin Filters in Xtraction.
- **Data model filter:** Determines whether the field can be included within Data Model Filters in Xtraction.
- **Date filter:** Determines whether the field can be included within Date Filters in Xtraction.
- **Default date:** Determines whether the field can be included within the Default Date Filters in Xtraction.
- **UTC date:** Determines whether the field is treated as a UTC Date field in Xtraction.
- **UNIX epoch:** Determines whether the field is treated as a UNIX Epoch Date field in Xtraction.

Click **OK** when finished. You will still need to save the data model file.

Table dialog > Fields tab > Expression Find & Replace

The **Expression Find & Replace** option is available by right-clicking in the Fields tab area of the Table dialog.



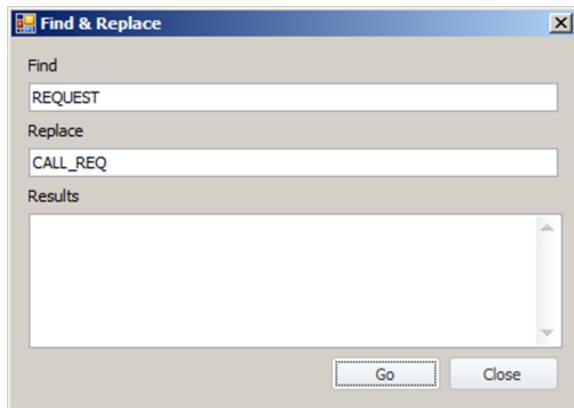
Use this feature to search for a specific string of characters in the Expression attributes of all fields within the table and replace those characters with a new string.

A practical use of this feature would be to change all references to a specific table or field in all field expressions within the table after renaming a table or field. (See example below.)

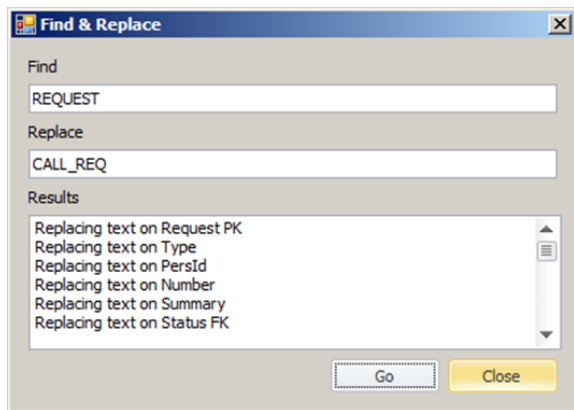
Another use would be to change all references to a specific database function or feature after copying objects from one type of database (e.g., Oracle) to another (e.g., SQL Server). For example, you could replace all occurrences of SUBSTR with SUBSTRING.

The example shown below assumes you're renaming the REQUEST table to CALL_REQ and want to update all field expressions accordingly.

Enter a Find string of **REQUEST** and a Replace string of **CALL_REQ**. Note the Find string is case sensitive.



Click the **Go** button to initiate the replacements, which are then displayed in the Results list.



The image below shows how the REQUEST string has been replaced with the CALL_REQ string in the Expression attribute for the Total Activity Time (hours) field.

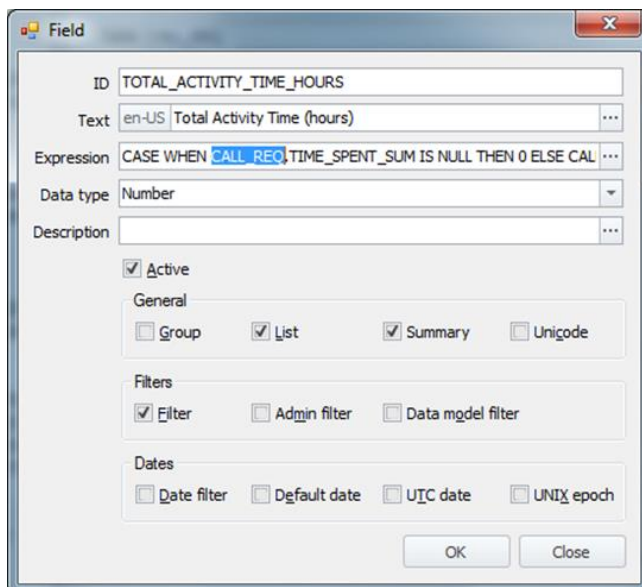
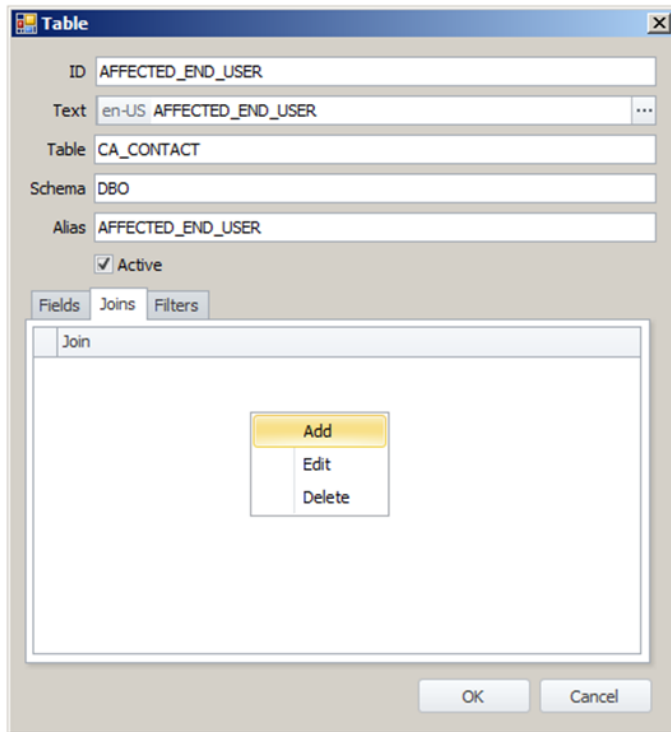


Table dialog > Joins tab

Use the **Joins** tab to add, delete, or edit table joins.

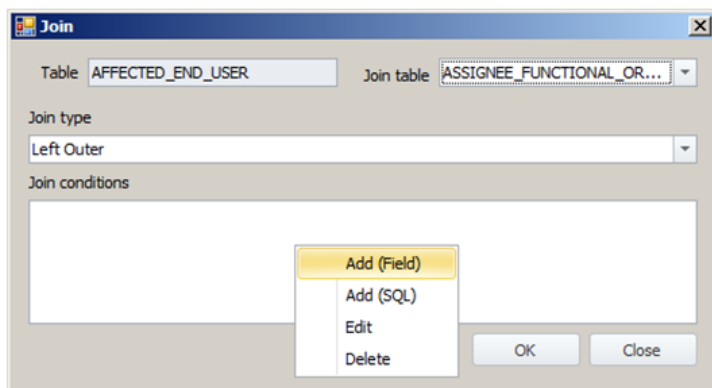


Joins are necessary when you need to retrieve data from other tables.

For example, the Affected End User field in the Request table may contain an identifier to a record in the Contact table. The First Name and Last Name fields are stored in the Contact table, along with other data.

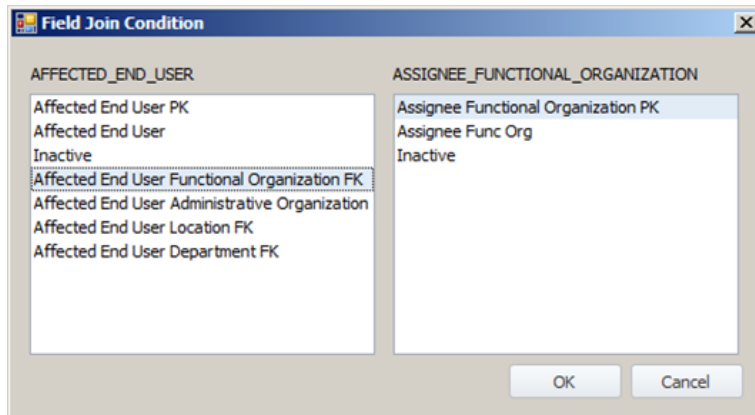
In these situations, a field like the Affected End User field is known as a **foreign key**, because it contains an identifier to a separate table. The identifier in the Contact table is known as a **primary key**, because it identifies reference data that other tables access via foreign key joins.

You can add an AFFECTED_END_USER table to the REQUEST table, as shown above. This table can be used to return the user's name and any other fields stored in the Contacts table.



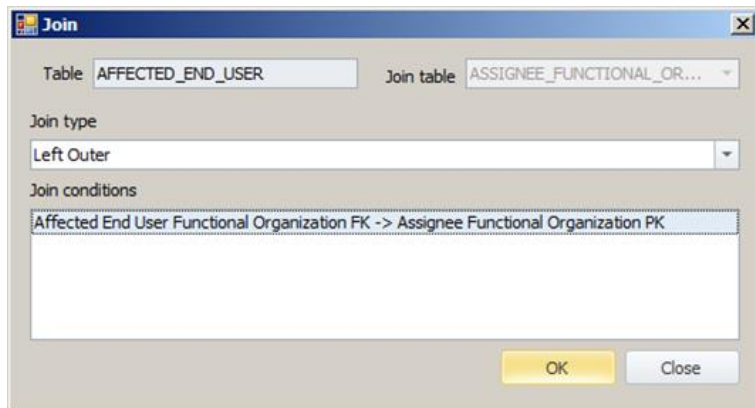
You can also add joins to this `AFFECTED_END_USER` table to return fields stored in other tables, like the Organization linked to the user. This example will be explained further as this join is constructed below.

When adding a new join, the details are entered in the Join dialog. By right-clicking and selecting **Add (Field)** you can add a field join, then select the fields to join together (shown below). Additionally, you can add SQL criteria to the join by selecting **Add (SQL)**.

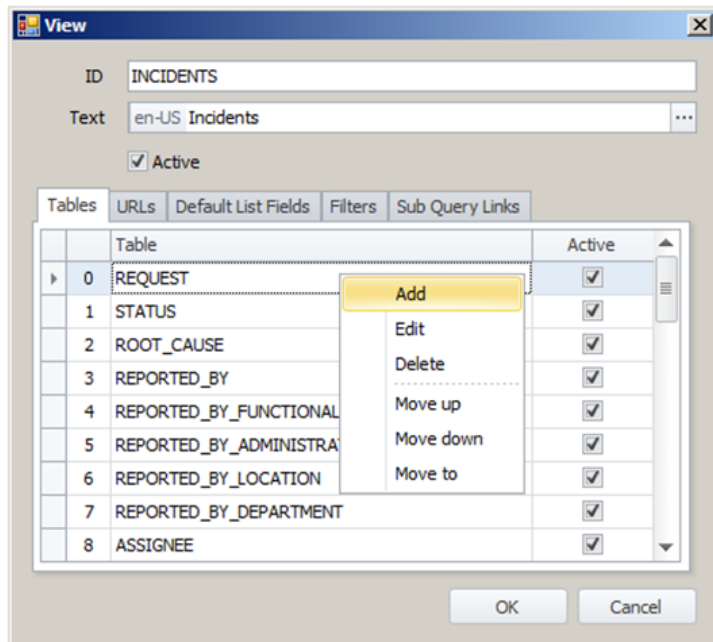


You need to select the Foreign Key field in the current table of interest (e.g., Affected End User Functional Organization FK in the above image) and the Primary Key field in the join table (e.g., Affected End User Functional Organization PK). By convention, the Foreign Key fields have a “FK” suffix added to the ID and Text attributes and the Primary Key fields have a “PK” suffix.

Click **OK** to build the join condition after selecting the two fields that define the join. The new join will be added to the Join conditions attribute, as shown below.

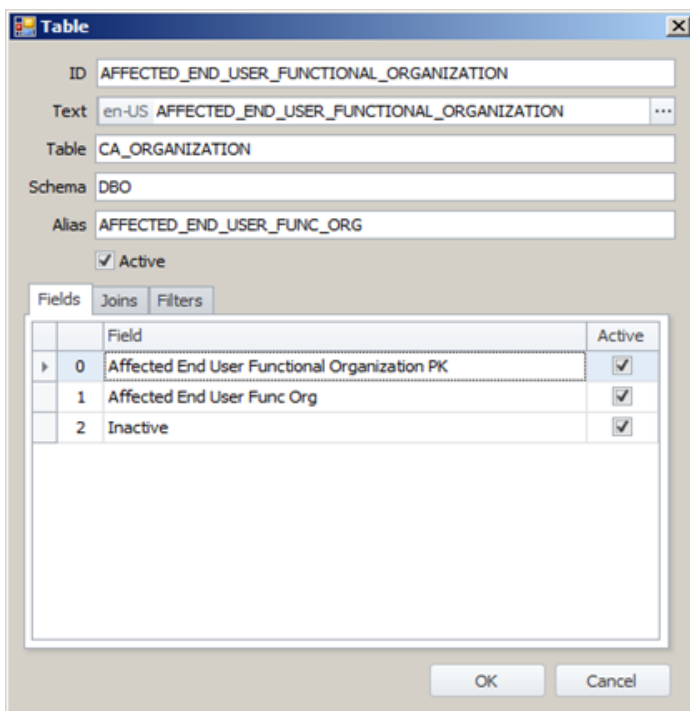


In this example, we’re defining a new join to retrieve the functional organization associated with the affected end user. To achieve this, we must first create a new join table, which is added to the View dialog > Tables tab for the INCIDENTS view, as shown below.



In our example, we have an **AFFECTED_END_USER** for an incident, and we want to retrieve the associated **FUNCTIONAL_ORGANIZATION** linked to that contact record.

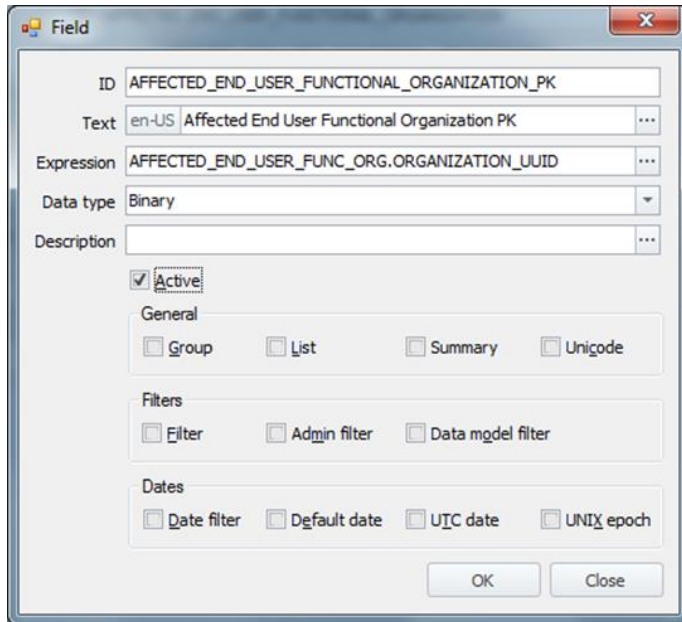
By convention, we name the **ID** and **Text** attributes of the new join table by concatenating the source field and the target field, all in uppercase, with words separated by the underscore character (i.e., **AFFECTED_END_USER_FUNCTIONAL_ORGANIZATION** in the image below).



The **Table** is CA_ORGANIZATION and the **Schema** is DBO. Notice the **Alias** is set to AFFECTED_END_USER_FUNC_ORG, which is a shortened version of the full name.

On the **Fields** tab, we can add fields to our join table. In our example, we have three fields, as shown in the image above. These fields are explained below.

The first field is typically the Primary Key field, which in our example has the **ID** attribute of AFFECTED_END_USER_FUNCTIONAL_ORGANIZATION_PK.



The **Expression** identifies how to access the field value (e.g., using the TABLE.FIELD syntax) and the **Data type** attribute indicates the type of data (e.g., Binary).

Since this is an internal UUID field, the only check box attribute we set is **Active**, because the field is used only to facilitate the join and doesn't need to be exposed to users.

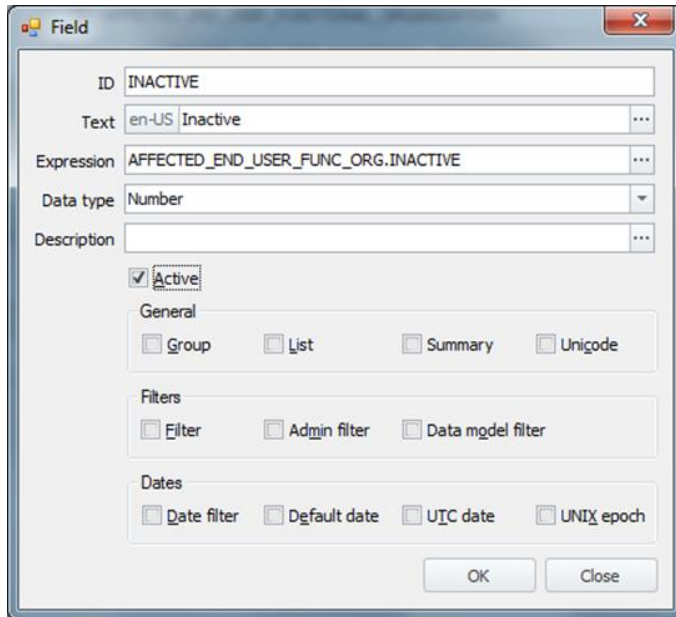
The second field in our example is the main target field, which has the **ID** attribute of AFFECTED_END_USER_FUNCTIONAL_ORGANIZATION.

The **Expression** identifies how to access the field value (e.g., using the TABLE.FIELD syntax) and the **Data type** attribute indicates the type of data (e.g., String). The **Description** is a text field used to display a description or tooltip in Xtraction when a mouse hovers over the column in various places, including a List Component column.

Since this is a target field exposed to Xtraction users, we set the check box attributes for **Active**, **Group**, **List**, and **Filter**, which are applicable to this particular String field.

Additional target fields can be added just like this one if we need to expose other columns accessible from the same database table. For example, while we’re retrieving the name of the organization via this join, we could also retrieve the associated telephone number, email address, or cost center, and expose these additional fields to users.

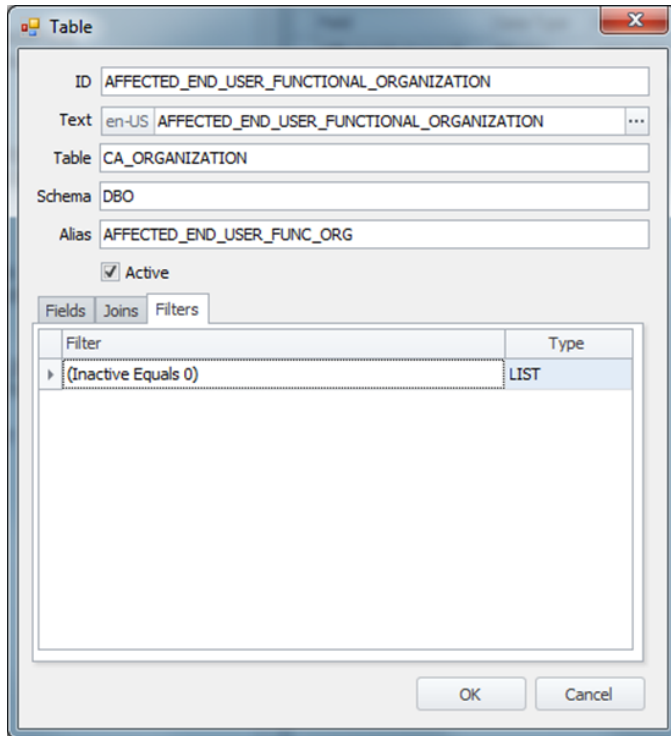
The third field in our example has been added to enable filtering. The **ID** attribute is set to INACTIVE, and the **Text** attribute is set to Inactive.



The **Expression** identifies how to access the field value (e.g., using the TABLE.FIELD syntax) and the **Data type** attribute indicates the type of data (e.g., Number).

Since this is an internal numeric field, the only check box attribute we set is **Active**, because the field is used only to facilitate the filter and doesn't need to be exposed to users.

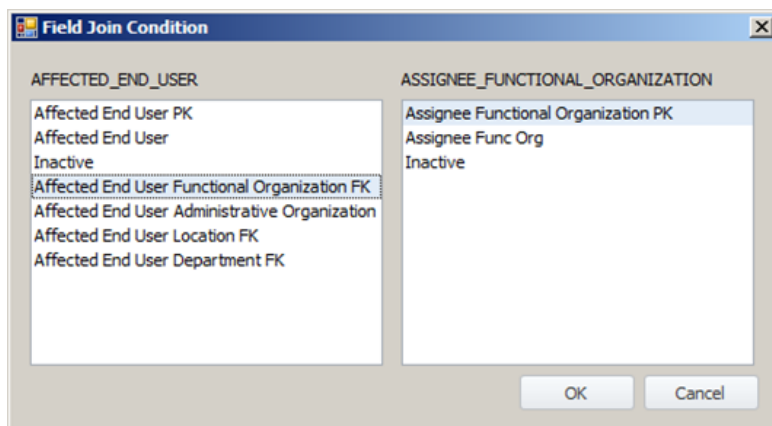
On the **Filters** tab of the Table dialog, we can add filters to our join table. In our example, we've added one filter. This filter uses the third field added above to filter out inactive records and retrieve only those records where Inactive equals 0.



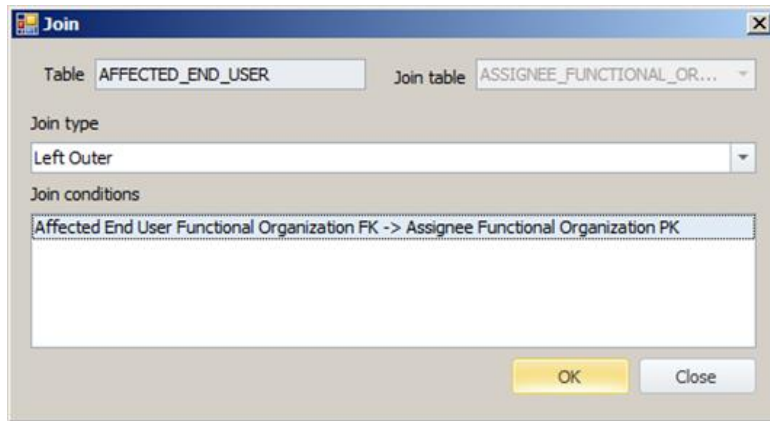
We could add additional filters, but if any additional database columns are required to define the filters, we must first add the corresponding fields to our join table.

Now that we have verified the `AFFECTED_END_USER_FUNCTIONAL_ORGANIZATION` join table has the required fields and filters defined, we can go back to our Join dialog for the new join we're defining on the `AFFECTED_END_USER` table, as started above.

We selected the Foreign Key field in the current table of interest (e.g., Affected End User Functional Organization FK) and the Primary Key field in the join table (e.g., Affected End User Functional Organization PK), as shown in the image below.



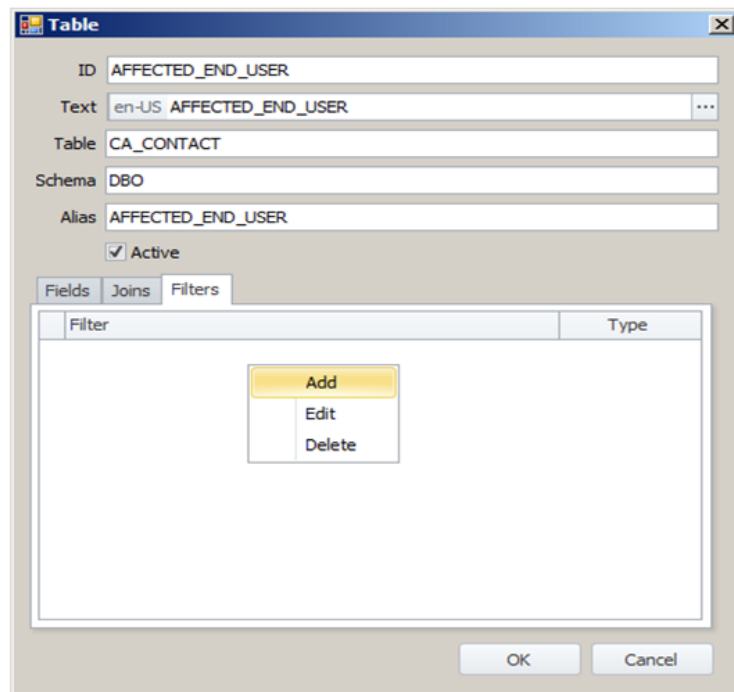
By clicking **OK**, the join condition is built, and the new join is added to the Join conditions attribute as shown below.



Next, we select the **Join type** from a drop-down list, then click **OK** to commit the changes (though the data model file still needs to be saved).

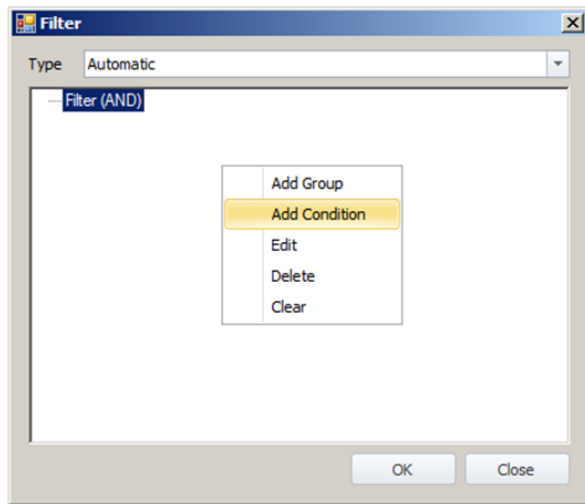
Table dialog > Filters tab

Use the Filters tab to specify filters that should be applied to the table when Xtraction retrieves record data. Right-click in the Filters tab area to add, edit, or delete filters.

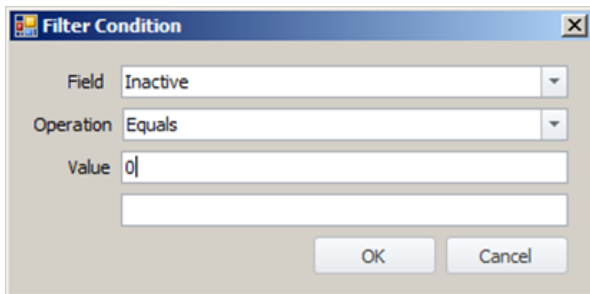


The Filter dialog displays if you're adding or editing a filter, as shown in the image below. You can then right-click in the Filter pane and select **Add Condition**.

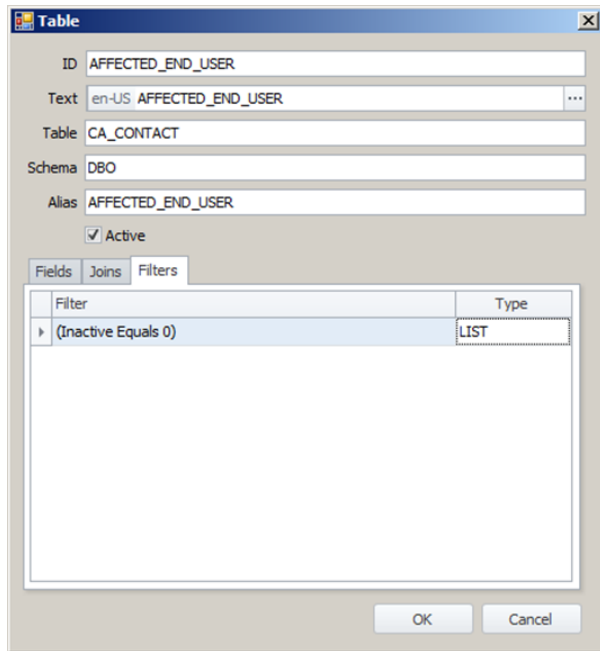
When defining a filter condition, you'll need to specify a **Type**, as shown in the image below. Automatic filters are always applied when Xtraction is retrieving data, whereas List filters are applied only when Xtraction is listing entries.



An example of the Filter Condition dialog is shown below.



Set the **Field** and **Operation** attributes from drop-down lists. Set the **Value** attribute according to the field values you want to filter in the record data returned by Xtraction. Click **OK** to commit your changes, though you'll still need to save the data model file. The new filter should be listed on the Filters tab of the Table dialog, as shown below.



Examples of complex fields

Most fields modelled for Xtraction are simple to define – enter an **Expression** attribute using straightforward TABLE.FIELD syntax (e.g., REQUEST.SLA_VIOLATION). This simplistic approach proves inadequate in some cases though, since the way some data attributes are stored in the database may not be the way users would prefer to see the values.

In some cases, it's useful to add a level of interpretation over the data to present information to users in a way that's easier to understand. To facilitate this, Xtraction allows the definition of complex field expressions using SQL functions and clauses.

The following examples demonstrate several methods of defining complex field expressions for Xtraction.

Binary Yes/No field

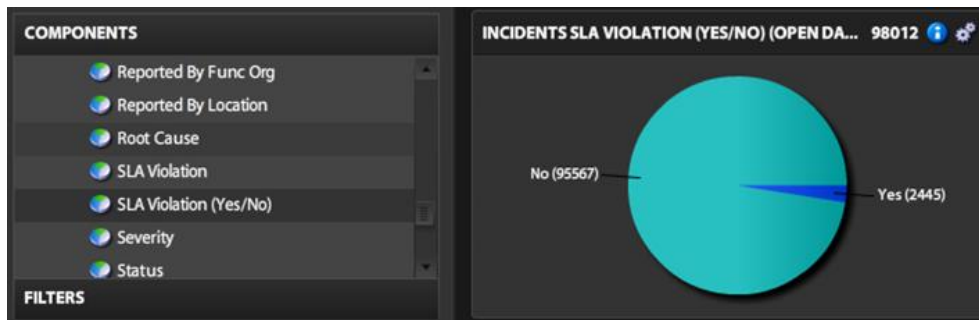
This example shows how to translate a Binary (0 or 1) field into a String field that displays either Yes or No.

In this case, we have a field in the REQUEST table called SLA_VIOLATION, which is a Binary field that contains either 0 or 1 (or NULL, if unset). We want to display the values as either 'Yes' or 'No,' so we define a new field called SLA_VIOLATION_YES_NO.

Instead of using the TABLE.FIELD syntax in the Expression attribute (which is how the SLA_VIOLATION field is defined), we use a CASE statement to translate the values. To view the full query text, click the ellipse button:

```
CASE WHEN REQUEST.SLA_VIOLATION IS NULL THEN 'No' WHEN REQUEST.SLA_VIOLATION = 0 THEN 'No' Else 'Yes' END
```

This example is for a SQL Server database. The result of using this field in Xtraction is shown below (the image is from an older version of Xtraction, but the result is similar in more recent versions):



Time field in hours

This example shows how to translate a Number field from seconds into hours.

In this case, we have a field in the REQUEST table called TIME_SPENT_SUM, which is a Number field that contains time recorded in seconds. We want to display the total activity time in hours, so we define a new field called TOTAL_ACTIVITY_TIME_HOURS.

Instead of using the TABLE.FIELD syntax in the Expression attribute (which is how the TIME_SPENT_SUM field is defined), we use the ROUND and CAST functions to translate the values. The full query text entered in the Expression attribute is as follows. This example is for a SQL Server database:

```
ROUND(CAST(CASE WHEN REQUEST.TIME_SPENT_SUM IS NULL THEN 0 ELSE REQUEST.TIME_SPENT_SUM/60.0/60.0 END AS FLOAT), 2)
```

The result of using this field in Xtraction is shown below (the image is from an older version of Xtraction, but the result is similar in more recent versions).

Number	Open Date	Status	Total Activity Time	Summary
SD:01	12/09/2002 4:00:00	Closed	0	Summary Service Desk Incident None
AM:11	12/09/2002 4:00:00	Closed	0	Asset Event ITIL Policy High
AM:12	12/09/2002 4:00:00	Closed	0	Asset Event ITIL Policy Medium
AM:13	12/09/2002 4:00:00	Closed	0	Asset Query ITIL Policy High

Time field in periods

In this example, we want to show how to translate a Date field into segmented periods.

We have two integer fields in the REQUEST table called OPEN_DATE and RESOLVE_DATE, which are Date fields that contain times recorded in seconds (i.e., the number of seconds elapsed since the initial epoch time of 00:00 on 01/01/1970).

To display the time to resolve in segmented periods, we define a new field called TIME_TO_RESOLVE.

Instead of using the TABLE.FIELD syntax in the Expression attribute (which is not that useful for these epoch date fields), we use a CASE statement to translate the values.

To view the full query text entered in the Expression attribute, click the ellipse button. This example is for a SQL Server database:

```

CASE
  WHEN REQUEST.RESOLVE_DATE IS NULL THEN '10. Not Resolved'
  ELSE
    CASE
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 600 THEN '01. < 10 Mins'
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 3600 THEN '02. 10 Mins - 1 Hour'
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 21600 THEN '03. 1 - 6 Hours'
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 86400 THEN '04. 6 - 24 Hours'
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 172800 THEN '05. 1 - 2 Days'
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 432000 THEN '06. 2 - 5 Days'
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 864000 THEN '07. 5 - 10 Days'
      WHEN (REQUEST.RESOLVE_DATE - REQUEST.OPEN_DATE) < 1728000 THEN '08. 10 - 20 Days'
      ELSE '09. > 20 Days' END
    END
END

```

Note the period labels contain additional spaces to improve readability.

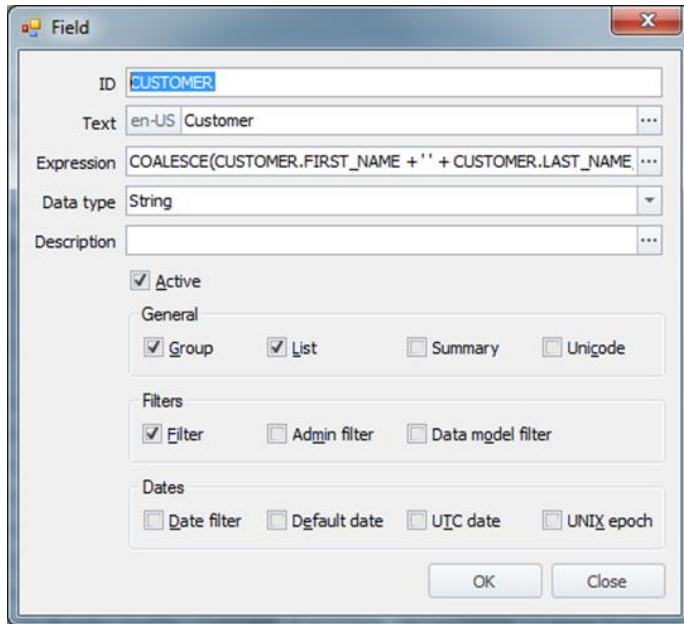
The result of using this field in Xtraction is shown below (the image is from an older version of Xtraction, but the result is similar in more recent versions).



Concatenation of String fields

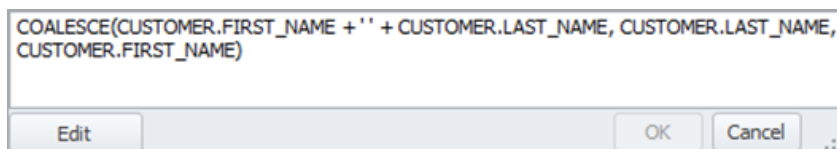
This example shows how to concatenate and manipulate String fields.

In this case, we have two String fields in the CUSTOMER table called FIRST_NAME and LAST_NAME. We want to concatenate the two strings, so we define a new field called CUSTOMER.



Instead of using the TABLE.FIELD syntax in the Expression attribute, we use the COALESCE function to manipulate the strings. The FIRST_NAME field string (if not empty) is concatenated with a space character and the LAST_NAME field string.

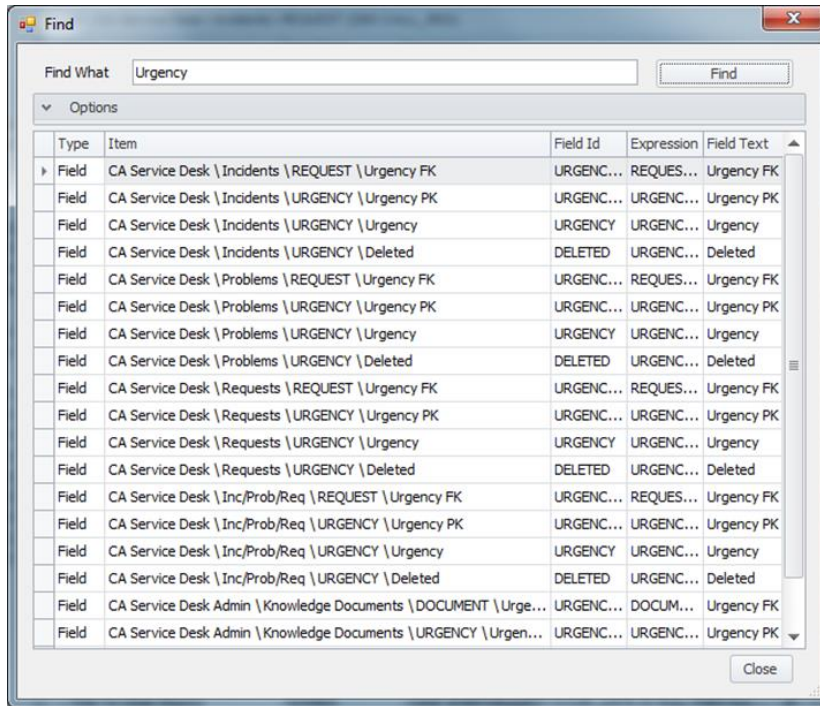
To view the full query text entered in the Expression attribute, click the ellipse button. This example is for a SQL Server database:



Data model Find & Replace

The data model Find & Replace feature is available from the Data Model Editor's **Edit > Find** menu. Enter a character string in the text box on the Find dialog and click the **Find** button. The default behavior is to find the entered text from all the available fields in either of the ID, Text, or Expression fields.

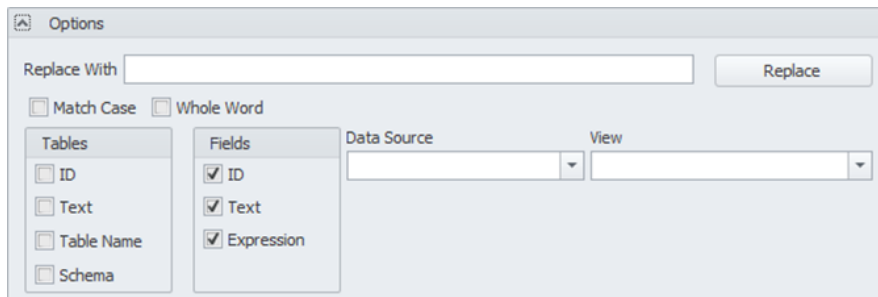
Find results are presented in a grid together with the fields included in the search. The following example uses the word "Urgency" as the find criteria.



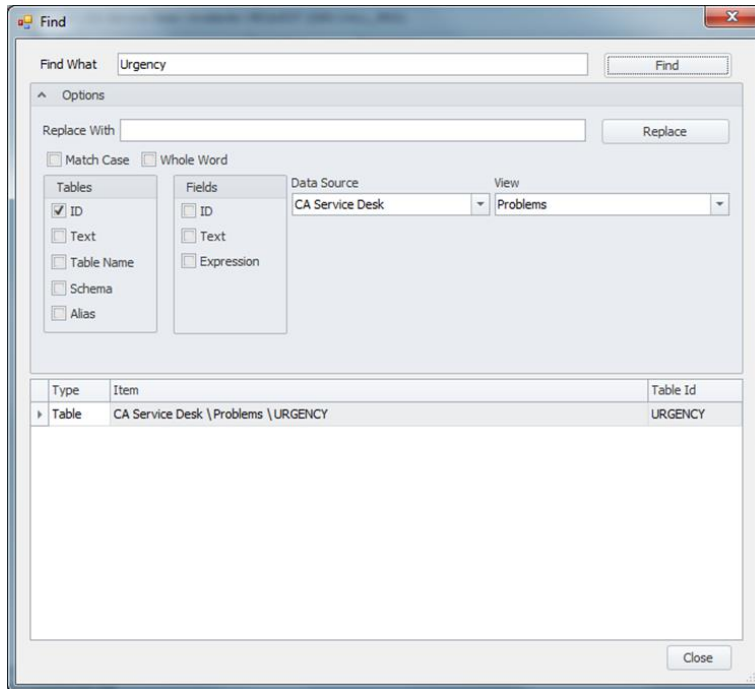
You can double-click any of the listed find results to open a Table or Field dialog for that specific object.

Additional Find options

Additional Find options are available by clicking the collapsible Options panel, as shown in the image above.



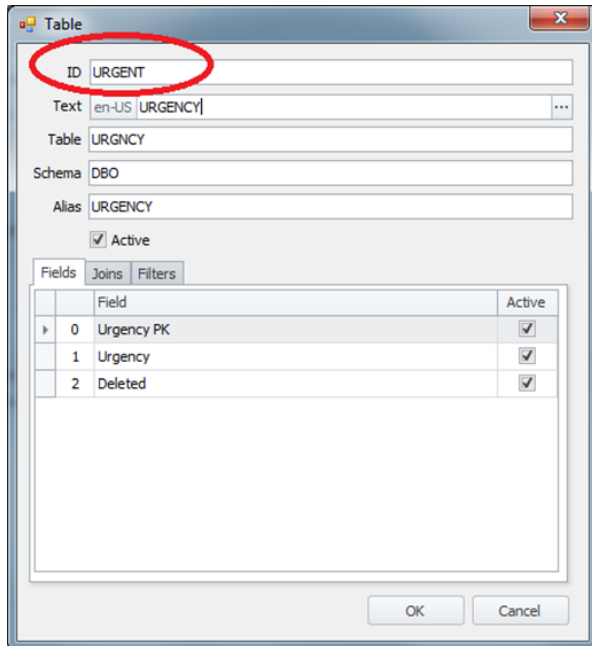
To narrow the above search down to a specific data source, view, and table, we can make the following changes:



If we need to change the Table ID to something else (e.g., URGENT) we can put “URGENT” in the Replace With text box and click **Replace**. The results from the replace are now shown in the grid:

Type	Item	Table Id
Table	CA Service Desk \Problems \URGENCY	URGENT

When you click an item, the underlying object is displayed, and you can see that the change is reflected in the table. Care should be taken when using Replace. **Remember to back up your data model before making any changes.**



Searching inactive references

As you modify a data model, you may want to make some tables or fields inactive. This is a reasonable thing to do. As of v2016.1, when you make a table or field inactive, there is a check against default list fields, table joins, URLs, and sub query links in case you've forgotten to remove any references to the table or field you're making inactive.

You may already have made tables or fields inactive without removing the references. In this case, there is a search feature to find all the references that use inactive tables or fields.

Open the Data Model Editor's **Edit > Search Inactive Reference** menu to view search results that are presented in a grid with the inactive table or field.

Type	Item	Table Id	Join Table	Field Text	Field Id	Details
DefaultFieldList	Zendesk With a lo...	GROUPS		Égroup	ÉGROUP	Field ÉGROUP is inactive in DefaultFieldList
Join	Our Service Desk ...	REQUEST	ROOT_CAUSE			Join Table ROOT_CAUSE is inactive
Join	Our Service Desk ...	REQUEST	REPORTED_BY			Join Table REPORTED_BY is inactive
Join	Our Service Desk ...	REPORTED_BY	REPORTED_BY_...			Join Table REPORTED_BY_FUNCTIONAL_ORGANIZAT...
Join	Our Service Desk ...	REPORTED_BY	REPORTED_BY_...			Join Table REPORTED_BY_ADMINISTRATIVE_ORGANI...
Join	Our Service Desk ...	REPORTED_BY	REPORTED_BY_...			Join Table REPORTED_BY_LOCATION is inactive
Join	Our Service Desk ...	REPORTED_BY	REPORTED_BY_...			Join Table REPORTED_BY_DEPARTMENT is inactive
Join	Our Service Desk ...	ASSIGNEE	ASSIGNEE_FUN...			Join Table ASSIGNEE_FUNCTIONAL_ORGANIZATION i...
Join	Our Service Desk ...	ASSIGNEE	ASSIGNEE_ADM...			Join Table ASSIGNEE_ADMINISTRATIVE_ORGANIZATI...
Join	Our Service Desk ...	ASSIGNEE	ASSIGNEE_LOC...			Join Table ASSIGNEE_LOCATION is inactive
Join	Our Service Desk ...	ASSIGNEE	ASSIGNEE_DEP...			Join Table ASSIGNEE_DEPARTMENT is inactive
Join	Our Service Desk ...	GROUP	GROUP_FUNCNTI...			Join Table GROUP_FUNCTIONAL_ORGANIZATION is i...
Join	Our Service Desk ...	GROUP	GROUP_ADMINI...			Join Table GROUP_ADMINISTRATIVE_ORGANIZATIO...
Join	Our Service Desk ...	GROUP	GROUP_LOCATI...			Join Table GROUP_LOCATION is inactive
Join	Our Service Desk ...	AFFECTED_END...	AFFECTED_END...			Join Table AFFECTED_END_USER_FUNCTIONAL_ORG...
Join	Our Service Desk ...	AFFECTED_END...	AFFECTED_END...			Join Table AFFECTED_END_USER_ADMINISTRATIVE_...
Join	Our Service Desk ...	AFFECTED_END...	AFFECTED_END...			Join Table AFFECTED_END_USER_LOCATION is inactive
Join	Our Service Desk ...	AFFECTED_END...	AFFECTED_END...			Join Table AFFECTED_END_USER_DEPARTMENT is ina...
Join	Our Service Desk ...	REQUEST	CONFIGURATIO...			Join Table CONFIGURATION_ITEM is inactive
Join	Our Service Desk ...	CONFIGURATIO...	CONFIGURATIO...			Join Table CONFIGURATION_ITEM_CLASS is inactive
Join	Our Service Desk ...	CONFIGURATIO...	CONFIGURATIO...			Join Table CONFIGURATION_ITEM_FAMILY is inactive

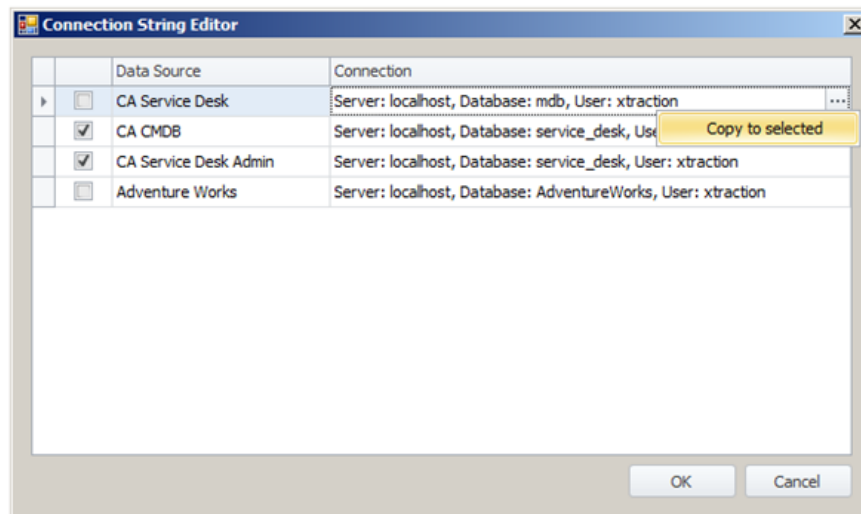
You can double-click any of the listed search results to open the View or Table dialog that gives you access to the default list fields, table joins, URLs, and sub query links associated with the search result.

Connection String Editor

A utility to edit the connection strings for each data source is available from the Data Model Editor’s **Tools > Connection String Editor** menu.

To edit the connection string of an individual connection, click the ellipsis button on the appropriate row and adjust the connection properties as required.

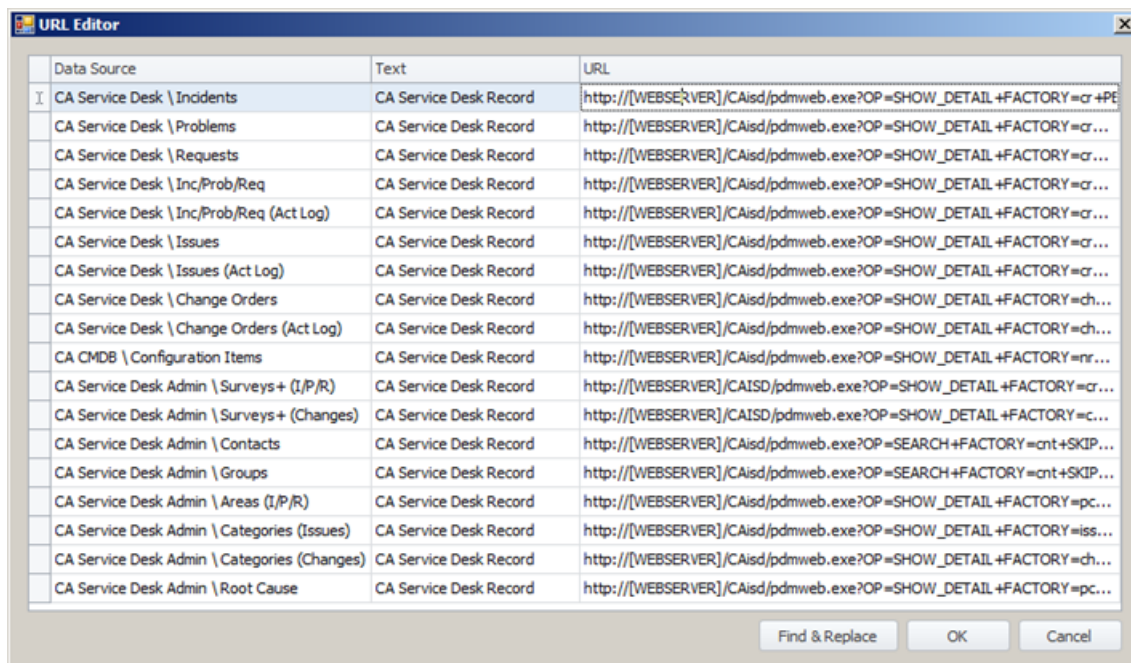
If you need to set the same connection string for multiple data sources, select the check boxes for the connection strings you want to overwrite. Then right-click the connection string to copy, and select **Copy to selected**, as shown in the example below.



URL Editor

The Data Model Editor provides a feature to globally edit the URLs recorded in a data model. To access this feature, open the **Tools > URL Editor** menu.

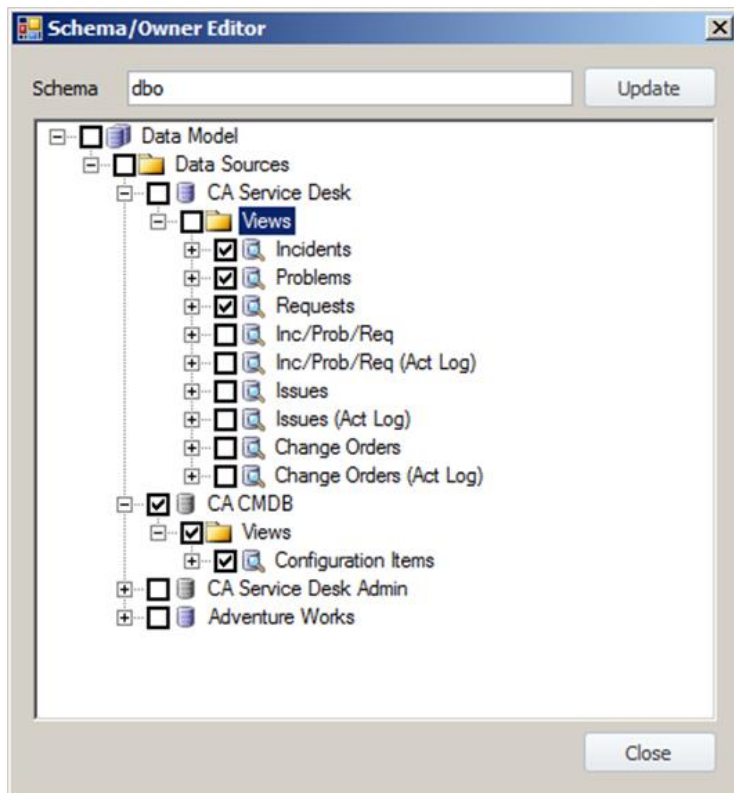
The URL Editor dialog displays, listing all the URLs defined for views within the data model. From here, you can edit the data model URLs individually for a particular view, or globally by using the Find & Replace button. Find & Replace is useful in a situation where the name of the service desk web server has changed.



Schema/Owner Editor

If the database owner/schema of multiple database objects changes, you can use the Data Model Editor to update the schema/owner of those items across data sources. You can apply updates at the level of data model, data source, view, or table. To access this feature, open the Data Model Editor's **Tools > Schema/Owner Editor** menu.

The Schema/Owner Editor dialog enables you to select which objects require a schema update. Click the check box next to the required object(s), enter the new schema name, then click the **Update** button as shown below.



Copying objects

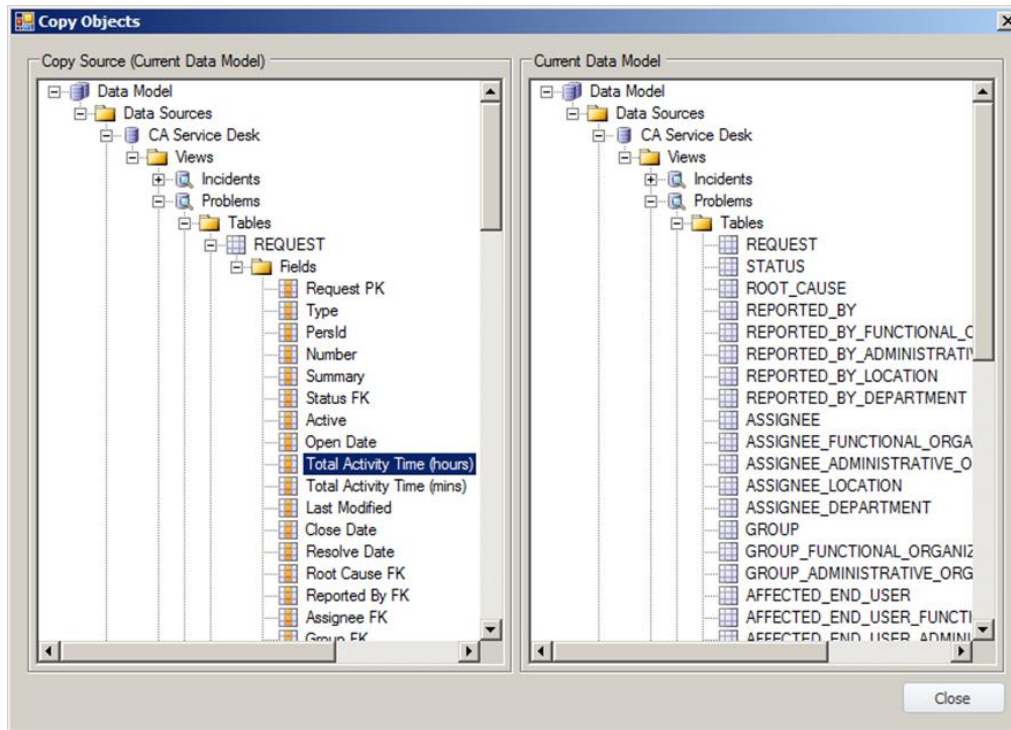
The Data Model Editor provides the capability to copy objects within or between data model files. This functionality is particularly useful when copying field definitions from one table to another, or when copying table definitions from one data model file to another. In fact, you can copy objects at any level including data sources, views, tables, and fields.

To access this feature, open the **Tools > Copy Objects** menu.

One important thing to note is that when you copy a **data source** or **view**, the included joins will be automatically copied because all required tables will be included in the copied views. When you copy a **table**, the included joins will not be automatically copied because some tables required for the joins may

not exist in the view. This means whenever you copy a single table rather than the whole view, any included joins must be copied or defined manually.

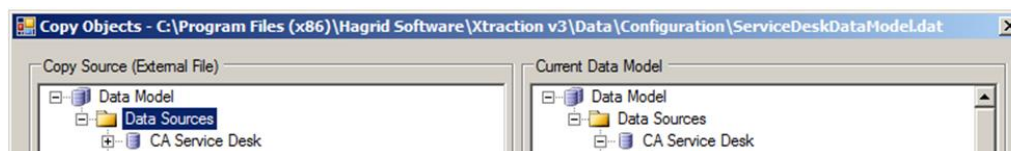
The Copy Objects dialog has two panes – a left pane for the source data model and a right pane for the destination data model – as shown in the image below.



The destination data model will always be the file currently open in the Data Model Editor. The copy source data model will default to the file currently open in the Data Model Editor. This means you can copy objects within the same data model file.

To copy from a different data model file, right-click in the Copy Source pane and select **Load External Data Model**. From the Open dialog, navigate to the data model file you want to open, and click the **Open** button.

When another file is selected for the source, it will display in the Copy Objects dialog title bar:



You can right-click in the Copy Source pane and select **Load External Data Model** to verify and/or change the source data model file selected.

Copy objects by dragging them from the Copy Source pane and dropping them into the desired container object in the Destination pane. For example, copy data sources to the blank area at the bottom of the Destination pane, copy views to the data sources, copy tables to the views, and copy fields to the tables.

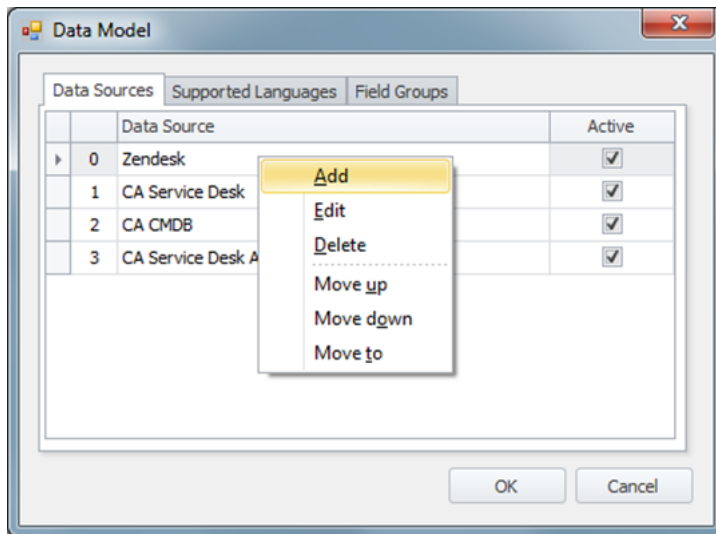
Click the **Close** button when finished, though you'll still need to save the updated destination data model file.

Saving a data model

You can save the current data model after viewing and editing by clicking Data Model Editor's **File > Save** menu. No dialog will display, and the data model file will be saved with the same name and to the same folder location, overwriting the existing file. To save the current data model to a different filename and or location, use **File > Save as**.

Data model settings

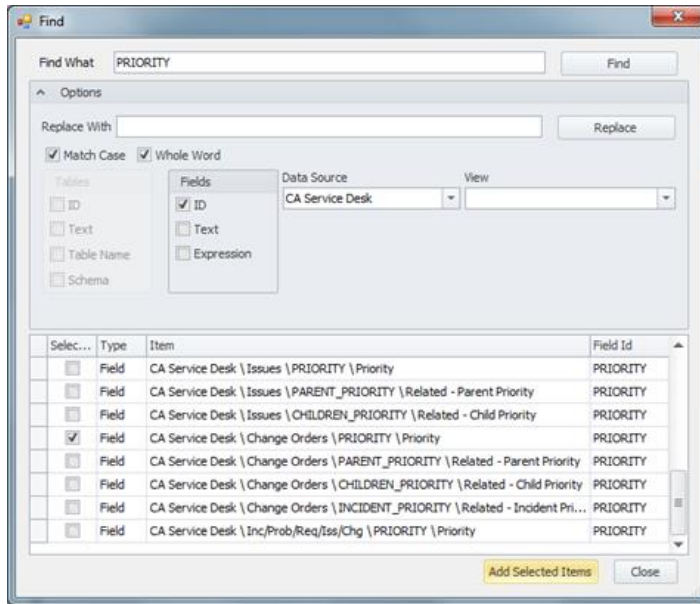
To open the Data Model settings dialog, double-click a data model in the Data Model pane. This dialog has three tabs. The **Data Sources** tab enables you to reorder data sources via the right-click menu, as well as access the Edit dialog for the selected data source.



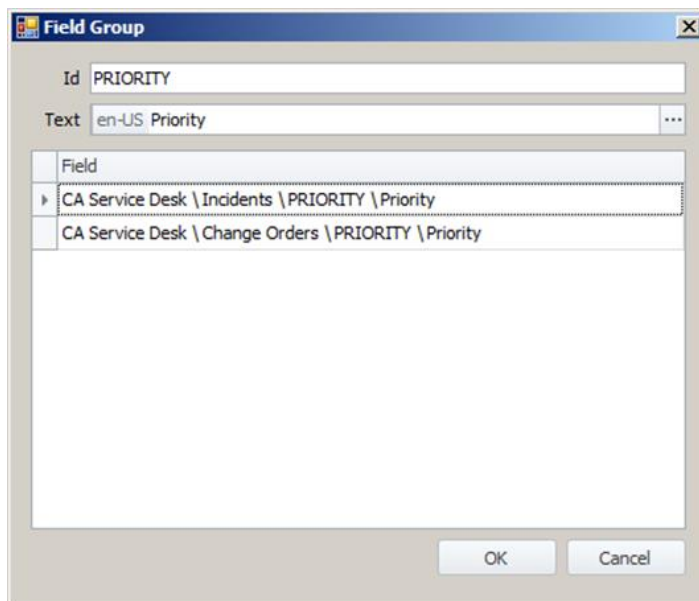
The **Field Groups** tab allows Xtraction to link associated fields together across multiple data sources and views. For example, consider the attribute Priority. If a Priority field is defined for both Incident and Change, specifying a field group for the Priority field allows Xtraction to dynamically filter components based on data from the Incident or Change view for the selected priority.

To add a field group, right-click in the Field Group pane and select **Add**. In the Field Group dialog, enter an **ID** and **Text** for the field group, and then right-click in the Field pane to add the fields you want grouped together.

In our example, we want to link **Incident Priority** with **Change Priority**, so we enter **Priority** in the search text box and click **Find**. We then select the check box next to CA Service Desk \ Incidents \ PRIORITY \ Priority and CA Service Desk \ Change Orders \ PRIORITY \ Priority, as shown below.



After selecting these fields, click the **Add Selected Items** button to add the fields to the field group.



Data model Schema View

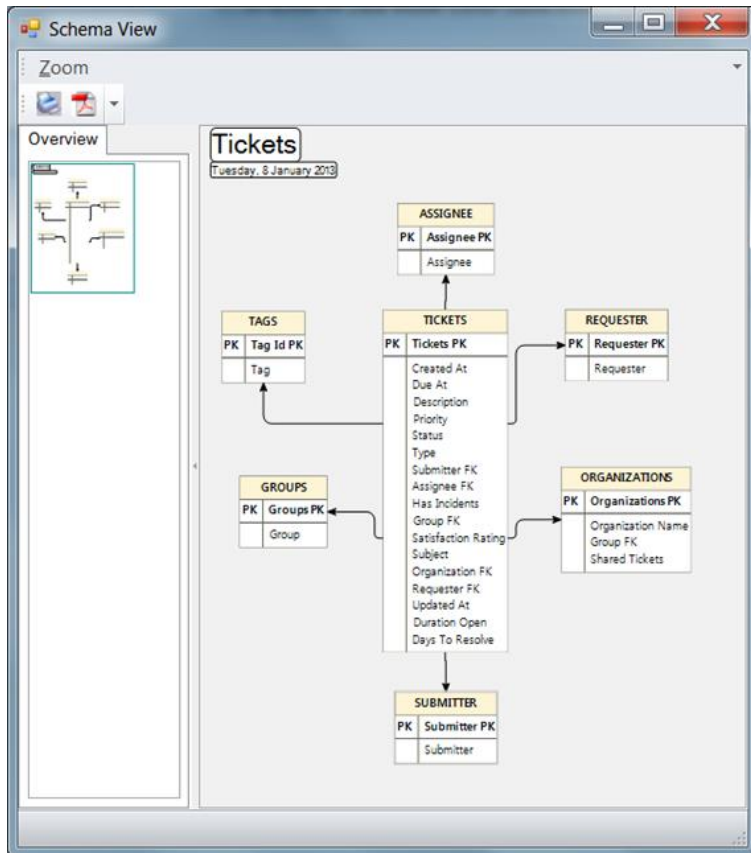
From version 13.1 of the Data Model Editor, a data model view can be displayed as a diagram, known as the Schema View. To display the diagram for a data model view, right-click the view from the tree and select **Show Diagram**.

The diagram represents the structure of the data model, not the underlying databases. It displays all tables within the view with their primary keys highlighted (provided they follow the recommended convention

with the **PK** characters added to their text labels). Table joins are displayed as lines and arrows between tables.

To edit tables, double-click the table name to open the table editor. To edit joins, double-click a connector line to display the join editor.

Use the Zoom menu to magnify the diagram. Below the Zoom menu is a two-item tool bar that enables you to print or export the diagram to PDF. To navigate around large diagrams, use the collapsible Overview panel on the left. Click inside the panel to move the central focus of the main diagram.



Example data model

This example shows the steps involved in modelling a Sales Order from the Adventure Works database.

1. Add a data source

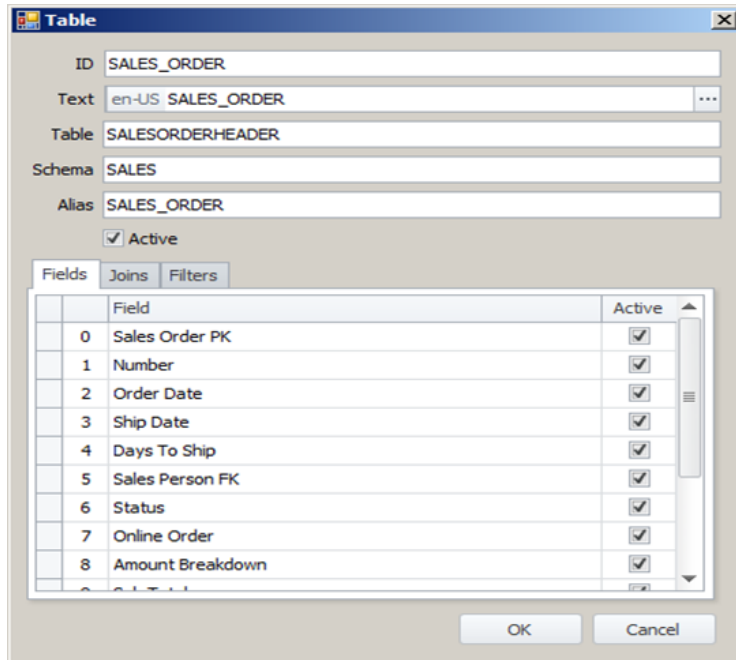
The first step is to add a new data source to a data model called **Adventure Works**. Be sure to provide a meaningful ID, as this is used through Xtraction to refer to the data source and should not be changed once set up.

2. Add a view

Next, add a new view called **Sales Orders** to the Adventure Works data source.

3. Add the Sales Order table

Add the relevant tables and fields for a sales order. Add the first table from the Adventure Works database called **SalesOrderHeader**.



Some of the modelled fields are:

- **Order Date:** As sales orders are normally filtered via the order date, this field is flagged as the Default Date.
- **Days to Ship:** Different types of SQL expressions can be used for fields. **Days to ship** is a calculated field showing the number of days between the order date and the ship date.
- **Sales Person FK:** The ID of the sales person stored against the sales order needs to be modelled so that the relevant sales person record can be included.
- **Status:** In the SalesOrderHeader table, the Status field is a number that maps to several status values not stored in the database. So that the data presented to the end user is meaningful, a SQL expression is used:

```
CASE SALES_ORDER.STATUS WHEN 1 THEN 'In process' WHEN 2 THEN 'Approved' WHEN 3 THEN 'Back ordered' WHEN 4 THEN 'Rejected' WHEN 5 THEN 'Shipped' WHEN 6 THEN 'Canceled' ELSE 'Unknown' EN
```

- **Online Order:** So that the data presented to the end user is meaningful, this field also uses a SQL case statement:

```
CASE SALES_ORDER.ONLINEORDERFLAG WHEN 1 THEN 'Yes' ELSE 'No' END
```

- **Amount Breakdown:** In this case, the total amount due is broken down into groups:

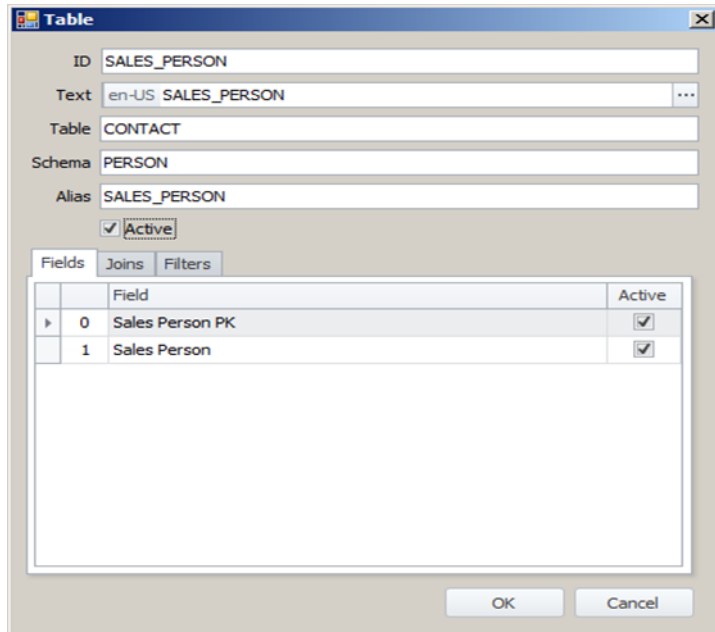
```
CASE WHEN SALES_ORDER.TOTALDUE < 500 THEN '01. < 500' WHEN SALES_ORDER.TOTALDUE < 1000 THEN '02. < 1000' WHEN SALES_ORDER.TOTALDUE < 2500 THEN '03. < 2500' WHEN
```

```
SALES_ORDER.TOTALDUE < 5000 THEN '04. < 5000' WHEN SALES_ORDER.TOTALDUE < 10000 THEN
'05. < 10000' ELSE '06. 10000+' END
```

- **Sub Total:** This field is flagged as Summary so that additional functions can be applied to it.

4. Add the Contact table (sales person)

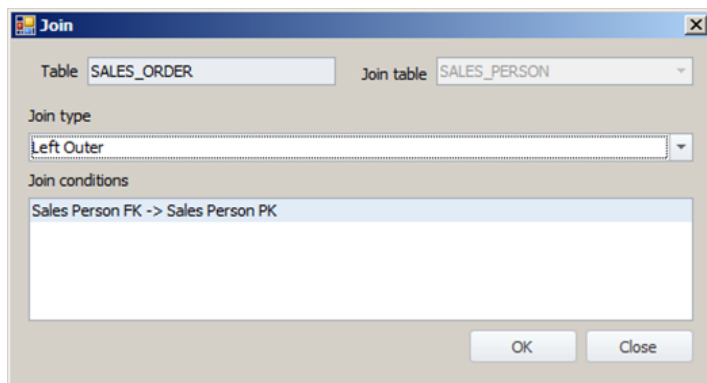
Model the sales person. Within the physical database, the join flows SalesOrder > SalesPerson > Employee > Contact. This example only requires details from one modelled table, the Contact table. Because we're modelling the sales person relationship in this instance, we keep all IDs/Text relevant to the relationship.



ID	Field	Active
0	Sales Person PK	<input checked="" type="checkbox"/>
1	Sales Person	<input checked="" type="checkbox"/>

5. Join the sales order to a sales person

The final step is to add a relationship so that Xtraction knows where the sales person details come from. For this, we need to go back to the Sales Order table to add it. Because a sales person doesn't need to be associated with a sales order in the Adventure Works database, we use a left outer join for the relationship.



6. Load the data model into Xtraction and view the results

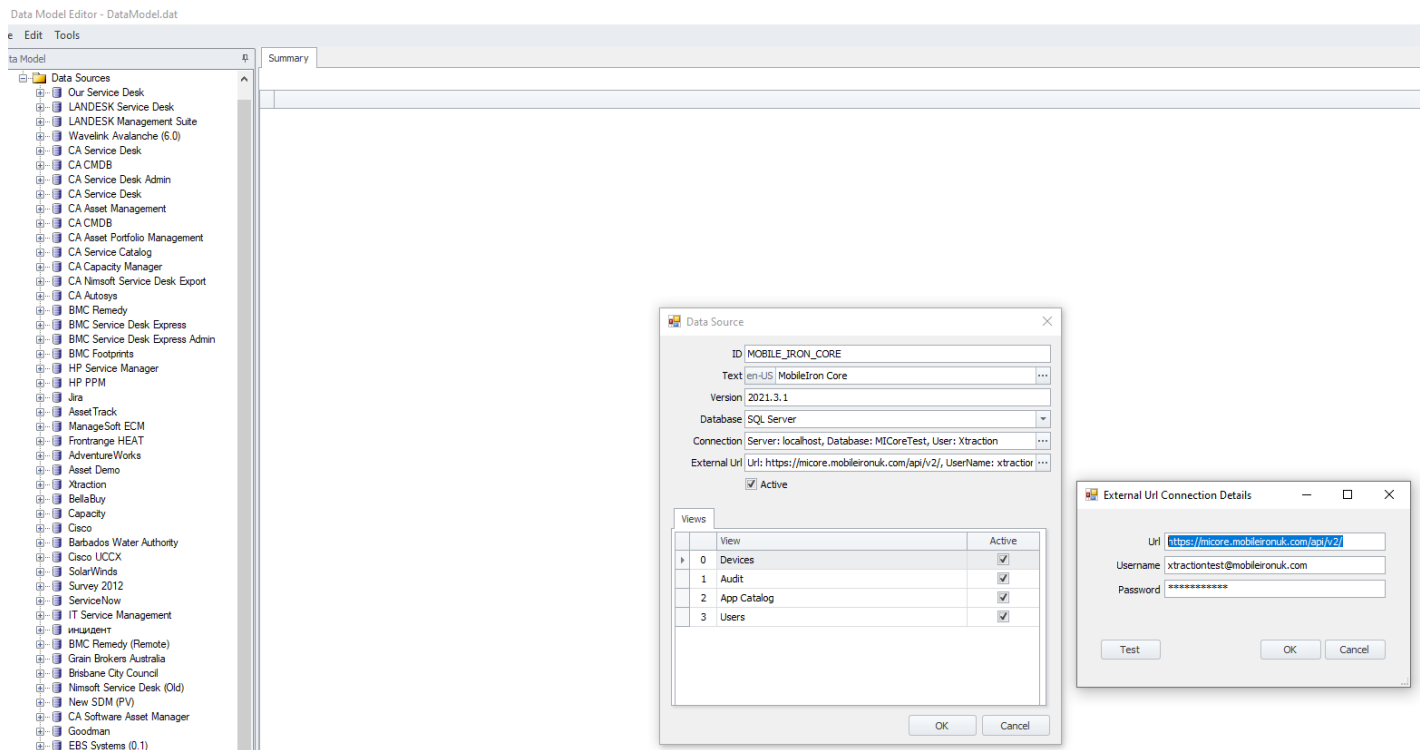
Adding MobileIron API calls to Xtraction

Xtraction 2022.1 enables you to configure, as part of the data model, API calls specifically for use with the MobileIron connector. These actions display in the Xtraction web interface and are available to execute from record lists. Upon selecting an action, an in-context message will display the results of the call (success or failure).

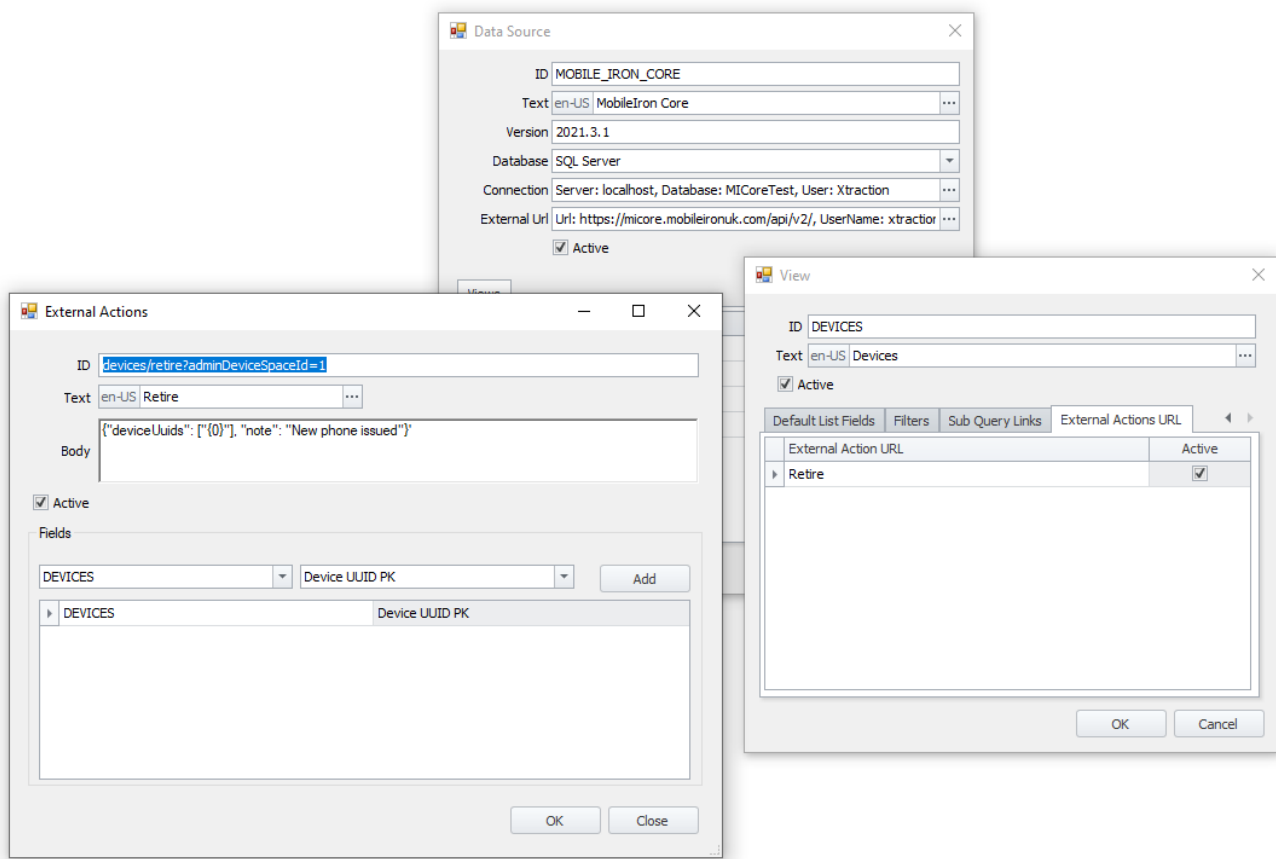
Use the Data Model Editor to edit your DataModel.dat file and add the API calls.

With the editor tool open:

1. Add the MobileIron connector to the **DataModel.dat** file.
2. Add the MobileIron URL **connection details**.

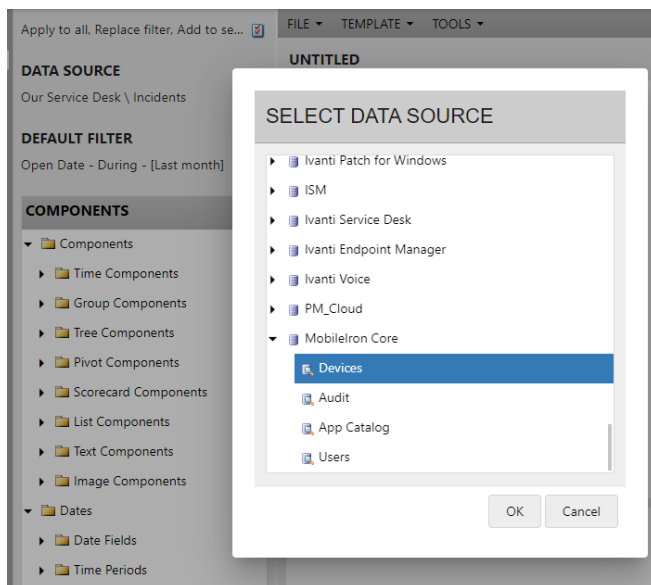


3. Add the API action on the desired **View** (in this example, it's added on **Devices** as shown below).



To see the results in Xtraction:

1. In the Xtraction web interface, select the **MobileIron Core > Devices** data source.



2. Create a dashboard with a Record List component.
3. Right-click a row to see the action (in this example, the action is to **Retire** the device).

Apply to all, Replace filter, Add to se...

DATA SOURCE
MobileIron Core \ Devices

DEFAULT FILTER
No filter defined

COMPONENTS

- Components
 - Time Components
 - Group Components
 - Tree Components
 - Pivot Components
 - Scorecard Components
 - List Components
 - Record List
 - Text Components
 - Image Components
 - Dates
 - Date Fields

UNTITLED

DEVICES

Display Name	Current Phone Number	Device Name	Manufacturer	Model	Platform	Home Country	Status
Florian S	000000000000	iPad (2)	Apple	iPad Air 2	iOS 15.4	United Kingdom	Active
Benjamin R	PDA 7		samsung	SM-G965F	Android 10.0		Active
Florian S	PDA 17		samsung	SM-A426B	Android 12.0		Active
Florian S			samsung	SM-A426B	Android 12.0		Active
Benjamin R	PDA 5		samsung	SM-G965F	Android 10.0		Active
Benjamin R	000000000000	iPhone	Apple	iPhone 12	iOS 15.0	Germany	Active
Xtraction Test	PDA				Android		Retire