Databridge Administrative Console User Guide

7.0 Service Pack 1

Table of contents

Databridge Administrative Console 4

Tour the Administrative Console 5

- Administrative Console Working Areas 5
- Dashboard 6
- Servers 6
- Authentication 6
- Documentation 6
- About 6

Getting Started 7

- Setting up the Administrative Console 7
- Add a Client Manager Service 8
- Connect to the Service 8
- Add a New or Existing Data Source Dialog 11
- Setting up the global configuration properties 11
- Creating a data source 12
- Generating scripts 12
- Cloning DMSII data sets and tracking changes 12

Servers Page 13

- View the Read-only Server Properties 13
- Add a Server 14
- Remove a Server 14

Dashboard 15

- Dashboard Columns 16

Setting up Authentication 20

- Setting up LDAP 20
- Creating Userid/Password Pairs 22

Client Manager 23

- Client Manager 23
- Service Trace Options 24
- Switch Service Logging 25

Manage Users 26
Set and Change the Console Password 29
Client Manager Information (Read-Only Properties) 30
Working with Data Sources 31
Working with Datasources 31
Add 38
Settings 45
Data Source Run Commands 60
Data Source Actions 63
Advanced Options and Commands 66
Customizing Data Sets 83
Customizing Data Sets 83
Data Set Properties 88
Data Set Information (Read-Only Properties) 95
DMS Items Properties 101
Customize DMS Settings 110
Data Items Properties 115
Data Item Information (Read-Only Properties) 120
Data Table Properties 122
Data Table Information (Read-Only Properties) 125
Global Parameters 127
Global Parameters 127
Bulk Loader 128
Customizing 132
Logging 163
Processing 168
PCSpan 193
Encryption 201
Legal Notice 202

1. Databridge Administrative Console

Use the Administrative Console to manage all Client operations. For instance, you can start, stop, and schedule Client runs and query Client runs to get status information and statistics.

Client output messages are displayed in the console pane for the data source. You can update the settings maintained in the Client and Client Manager Service's configuration files and customize the layouts of the tables in the relational database that hold the replicated DMSII data.

For an introduction to the graphical user interface, Tour the Administrative Console.

In addition, the Administrative Console provides a single window from which you can monitor multiple data sources and/or Client Managers, while also receiving alerts when a Client or Client Manager has encountered an issue.

Reference

See the *Databridge Client Administrator's Guide* for a complete reference to the Databridge Client, including configuration settings, user scripts, and command-line Client operations. The PDF guide is also included with the Databridge software (client-admin.pdf).

2. Tour the Administrative Console

The Administrative Console was updated to a browser-based console for the Databridge 7.0 release. The new Administrative Console allows authorized users to

- manage Databridge Client **Process** and **Clone** command runs
- customize data sources
- monitor client runs and Client Manager services.

You can accomplish each phase of the replication process by using menu commands and options.

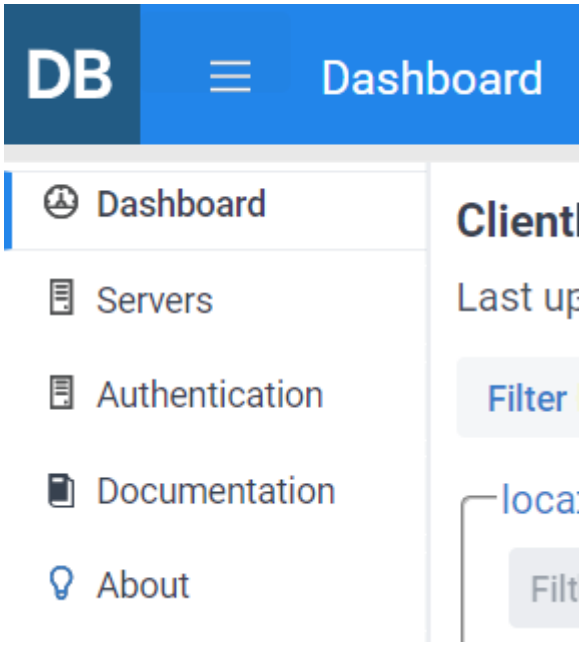
This tour highlights the main areas of the Administrative Console.

2.1 Administrative Console Working Areas

The Administrative Console has distinct working areas that are divided into multiple groups of pages displaying different types of information and configuration options. At the top level of the Administrative Console, select the menu icon ( ) to expand the left-side navigation.

Left Navigation Menu

Here are the options on the left navigation menu. Each area is described below.



2.2 Dashboard

The Dashboard provides a customizable table view of status information, properties, and statistics for **Process** and **Clone** command runs controlled by the various Client Managers.

For more information, see the Dashboard page.

2.3 Servers

The **Servers** page lists all the servers known to the Administrative Console. These servers are saved in a file, which gets read when the Administrative Console server is started. The file is updated when a server is added or removed from this list. The **Dashboard** page uses this list to find all the Client Managers that are monitored.

For more information, see the Servers page.

2.4 Authentication

The Authentication page allows administrators to setup authentication to the Administrative Console. Administrators can either configure **LDAP**, or create userid/password pairs when using **Simple Authentication**. Users other than the System Administrator (dbridge) are only allowed to change their own passwords using this left menu item when **Simple Authentication** is being used. The other actions in the dialog are not allowed for users.

The System Administrator can configure LDAP, add, modify and remove users when using Simple Authentication, and change any user's password. For detailed information on setting up and configuring the Administrative Console, see the Authentication page.

2.5 Documentation

Opens the Administrative Console Help in a different browser tab.

2.6 About

This is the About page for the Administrative Console that displays the version of the software and other relevant information.

3. Getting Started

This section provides instructions for connecting to the Client Manager service on the Client system and adding data sources.

> **Note**
>
> These terms are used throughout this guide:
>
> - *service* refers to both the service (Windows) and the daemon (UNIX).
> - *Databridge Server* refers to both Databridge Server (DBServer) on the host and Databridge Enterprise Server (DBEnterprise) on a Windows computer.

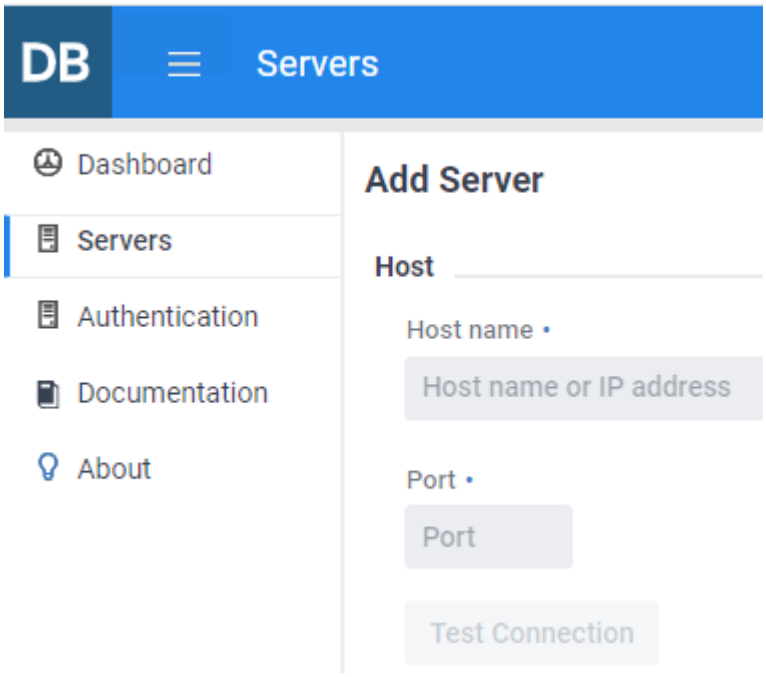
3.1 Setting up the Administrative Console

After installation, the first connection to the Administrative Console must configure the **Authentication** settings. The options are **Simple Authentication** or **LDAP**.
Simple Authentication requires that all userid/password pairs be added to the file that maintains this list (passwords in the file are encrypted).

For more information on setting up **Authentication** for the Administrative Console see the Authentication section of this guide or consult the *Databridge Installation Guide*.

After completing the Authentication setup, add the Client Manager service(s) to the list of servers as explained in the Add a Client Manager Service section below. Once the Client Manager(s) are added, the file will be read when the service starts, and it is preserved when a Hot Fix, Update, or Service Pack is applied.

3.2 Add a Client Manager Service



Open the left navigation menu from the ( ) icon in title bar and click **Servers**, which opens the **Servers** page.

Make sure that the Client Manager service is running on the Client system. (For instructions, see the *Databridge Installation Guide*.)

To add a Client Manager Service:

1. Click (Add).
2. Enter the **Domain Name** or **IP address** of the Client Manager and the port number (typically 8001).

   The console will auto-detect the type of server.
3. Once the server is verified, it will appear in the list of servers on this page as **Client Manager**.

   This information will be preserved.

3.3 Connect to the Service

After a Client Manager is added to the list of servers, navigate to the data sources page. As a result, the console session automatically connects to the Client Manager service. From the **Servers** page, click the name or IP address of the Client Manager, which is a link to the corresponding data sources page.

From the **Dashboard** Page, which is the default page a console session starts on, select the link in the upper left corner of the group of data sources for the Client Manager to get to the data sources page for the Client Manager.

If the Client Manager is already configured, which happens when you upgrade from an earlier release, all the data sources configured in the service will appear on the screen.

When you have multiple Clients on the *same* **machine**, you can manage them with a single service. The service controls all client runs and the **DBClntCfgServer** program. The **DBClntCfgServer** program is a specialized Client that provides access to the relational database for the service and performs all Client operations initiated by the Administrative Console (except for `Process` and `Clone` commands). It also operates cooperatively with the **Customize** command, which allows you to perform customizations that control how the DMSII data set and data items get replicated to the relational database. When executing these commands, the program acts as a server that services the requests made by the user.

> **Tip**
>
> This is a simpler method of customization than working with user scripts. Creating SQL scripts is not required and the user will not have to work with the Client control tables.

When you have multiple client machines with Client Manager services, each Client Manager has its own data source page. You need to add each Client Manager in the **Servers** page to make them known to the Administrative Console.

For new installations, data sources must be added to the Client Manager service before you can use the console to configure the clients, start client runs, set up scheduling for regularly occurring client runs, and issue operator commands directed at the runs. For more information, see Schedule Updates.

3.3.1 To add a data source to a Client Manager service

> **Note**
>
> This step is only performed once. The console gets the list of data sources configured in the service with each connection.

From the Client Manager data source page, click **+Add** in the menu bar at the top of the page. The drop-down will have the following options:

- New (Add a new data source)
- Existing (Add an existing data source)

When upgrading from an older installation that was using the service, the update can be done with either of these options:

- Install the new the software using the same working directory as the older version (applies only to an upgrade from version 6.2 SP1 or newer). Start the service, and add the Client Manager(s) to the Administrative Console.

  > **Warning**
  >
  > If you have any questions regarding upgrading, please consult the *Databridge Installation Guide* and contact Micro Focus Customer Support .

- For a more cautious approach, create a new service with no data sources by specifying a new working directory during the installation.

  a. Copy the working directory of the data source into the new working directory and follow the process to Add a Data Source.

  b. Delete all obsolete files, including log files, discard files, and any leftover files in the root of the data source's working directory because these are present in the old working directory.

  c. Disable the data source in the older service, and use the Add Existing option to add the data source to the new service in Administrative Console.

  Once you determine that this data source is operating as expected, repeat the process for additional data sources.

> **Note**
>
> If you are running two services during the upgrade period, you will need to use a different port.
>
> For example, if the old service is using port 8001, you should specify port 8002 for the new service. The port number cannot be changed in the Administrative Console because a connection is required to run the service to be able to change the port.
>
> If you do not specify a port other than 8001 during the installation, you can use the **dbctrlconfigure** `export` command to create the text configuration file `dbcontrol.ini` in the service's config directory. Update the port number in the file and run a **dbctrlconfigure** `import` command. Finally, restart the service to update the port number.

For a new installation, the **+Add** > **New** option is recommended. In order to add a data source, the server to which the data source will connect needs to be added to the Servers list. If the server is not present in the drop-down list, use the Add New Server dialog to manually add the server to the Administrative Console. The Administrative Console will automatically detect whether the server is a Databridge Enterprise Server, or, a DBServer running on an MCP system. In the case of DBServer, it will also detect if the connection to the server is using TLS.

> **Important**
>
> Currently, only DBServer supports encryption in this release.

For previous command-line Client users, add your existing data sources to the Administrative Console provided that their working directories are subdirectories of the Client Manager's working directory.

Once a data source is added to the service, it sends all Administrative Console sessions (including the monitor) an event notification confirming the addition of the new data source.

Any new Administrative Console session, upon connection to a Client Manager, issues a Remote Procedure Call (RPC) to get the list of data sources that the service manages. This list will contain the data source that was added.

3.4 Add a New or Existing Data Source Dialog

See the relevant topics for information on Adding Data Sources:

- Adding a New Data Source
- Adding an Existing Data Source

3.5 Setting up the global configuration properties

Once you add a data source, the service will create its working directory and configuration file. The configuration will have all global parameters set to their default values, except for the signon section parameters that are set to the values specified in the **Add New Data** dialog.

After you add a data source, we recommend that you update the client configuration parameters. This update will make customizations easier because the defaults will be correct most of the time and will reduce the number of changes you need to make. To do this, select **Settings** > **Configure** to change any parameters that you want to be changed globally.

> **Example**
>
> If you want to flatten all OCCURS clauses within the tables, set the corresponding parameter accordingly in **CUSTOMIZING** > Advanced page of the **Configure** command dialog.

3.6 Creating a data source

After the global configuration properties are set up, run either the **Customize** command from the **Settings** menu, or the **Define/Redefine** command from the **Actions** menu for the selected data source in the data sources view. These commands create the entries for the data source in Client control tables.

Typically, you will not want to replicate the DMSII data sets to the relational data base. The first customization you will perform is to exclude some data sets from cloning and tracking.
When using the **Customize** command, you can do this by updating the properties of the specific data sets by setting their **Active** properties to **Inactive** in data sets view of the **Customize** command. For more information on setting data set properties see the Data Set Properties page.

A **Define/Redefine** command is not as simple; first, create the Client table by running the command. Next, create a user script that sets the `active` column to 0 for all the data sets that you wish to exclude, then run the command again.

If you want to filter out data sets that contain confidential or sensitive information, consider using a logical database or a filtering routine in the DBGenFormat utility on the host. The Client has no record that these data sets exist. This method is just as effective as doing this in the client by setting the `active` columns to 0 for unwanted data sets. When running a **process** command data set whose `active` column is 0 are not selected. So there is no cost benefit to using either of these methods, the end result is the same.

3.7 Generating scripts

Once a data source is customized using either the **Customize** command or a **Define/Redefine** command the next step is to generate the scripts files that the client uses to create tables, indexes, and the files associated with running the bulk loader. To do this, select Generate Scripts in the **Actions** button drop-down found on the menu bar of the data sources page.

3.8 Cloning DMSII data sets and tracking changes

To clone a DMSII database, select **Process** from the **Actions** drop-down for the data source. This command instructs the Client Manager service to launch a **DBClient** run that executes a **process** command.

This run creates the tables and stored procedures (if applicable) for all the tables associated with the data sets whose `active` column is 1 in the DATASETS Client control table. (This is reflected in the Active property of the data sets.) To clone specific data sets, select the **Clone** command from the **Advanced** button drop-down for the data source.

4. Servers Page

Getting there

- Select the menu icon ( ) to expand the left navigation menu and select **Servers**

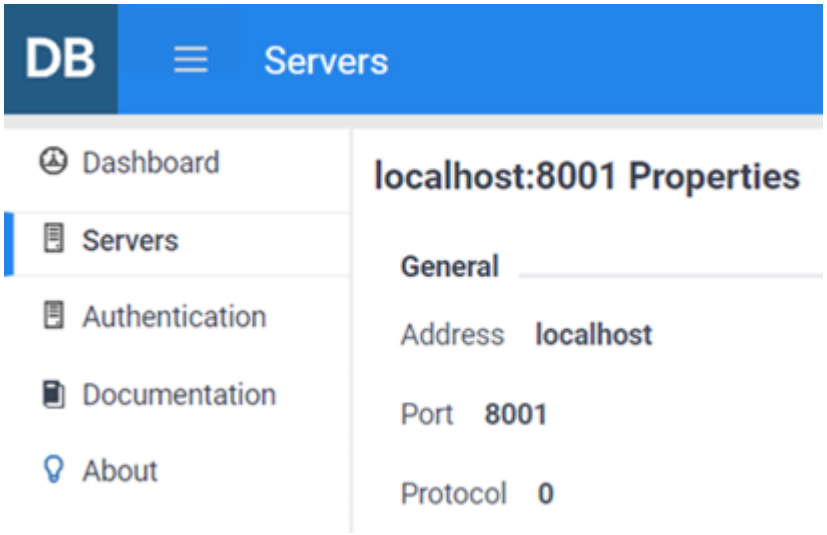
The **Servers** page lists all the servers known to the Administrative Console. These servers are saved in a file, which gets read when the Administrative Console server starts, and gets updated when a server is added or removed from this list. The dashboard uses this list to find all the Client Managers that are monitored.

From the **Servers** page, users (with access) can

- View the Read-only Server Properties for each listed Server
- Add a Server
- Remove a Server

4.1 View the Read-only Server Properties

From the Servers list, click the light bulb icon to view the Servers Properties.



The read-only properties for the selected Server display.

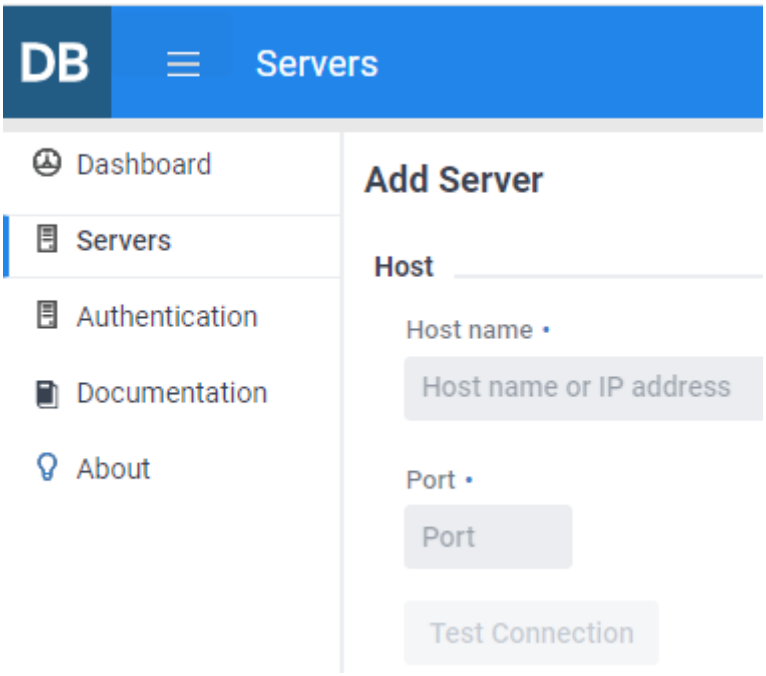
- **Address:** the **Host name** or **IP Address**.
- **Port:** the port number for the Server.
- **Protocol:** the protocol level.

4.2 Add a Server

To add a server to the Administrative Console:

1. Select the menu icon at the top left of the Administrative Console and open the **Servers** page.
2. From the **Servers** page click + Add and click **New**.

   The **Add Server** page displays.



3. Enter the **Host name** or IP address and the corresponding **Port** number. Click **Test Connection** to verify the entries.
4. Then click **Add** at the bottom right of the Administrative Console.

4.3 Remove a Server

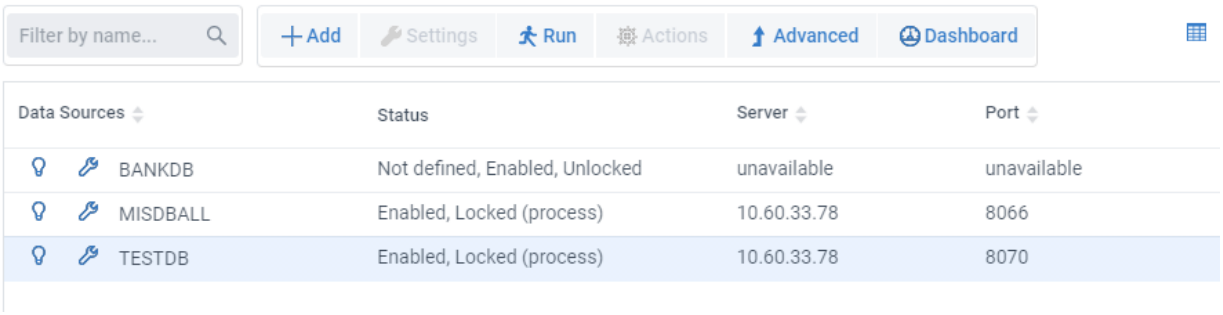
To remove a server from the Administrative Console:

1. Click the delete icon for the server you wish to delete.
2. Click **Yes** when prompted to confirm the action.

5. Dashboard

Getting there

The Dashboard loads when connecting to the Administrative Console. From the Client Manager's data sources page, click the Dashboard button.



The Dashboard (also known as the monitor page) provides a customizable table view of status information, properties and statistics for the various **Process** and **Clone** command runs controlled by the various Client Managers. The Dashboard also provides status information on the state of the Client Managers.

Whenever the Client generates an alert, it appears in the table as a warning icon () to indicate that there is a problem. Click the warning icon to display the associated message.

Statistics such as lag time, throughput, and the number of records processed can be viewed from the Dashboard. These customizations are global; they are remembered on subsequent monitor screens for all users until someone changes them.

The **Filter Columns** button allows the operator to customize which items to display. The monitor periodically polls all active runs, which respond with a statistics record that is used to update the display. The Administrative Console in turn reads the monitor data and refreshes the screen of the consoles on the Dashboard page. If a Client manager has no active runs, the monitor polls it to determine if it is alive.

If a service fails to respond to this request, the monitor issues an alert informing the operator that the Client Manager is not responding. The Dashboard is the control center for Databridge operations, as you can see all the Client Managers and data source on one screen. If you need to see more, you can navigate to the console for the Client Manager in question by clicking on the link (the name or IP address of the service) at the top left corner of the group of data sources for the Client Manager.

5.1 Dashboard Columns

The **Filter Columns** open a dialog that allows you to select the column(s) to include in the Dashboard tables. The columns are described below; options with an asterisk (`*`) are selected by default.

Run Type

The run type is `process` or `clone` .

Start time

This is the time at which the run started.

Stop time

This is the time at which the run ended, if the run is active, this column will be blank.

Lag time

The lag time is the difference between the time an update is applied to the relational database and the last timestamp encountered in the audit trail audit time stamp when the Engine processes the update (adjusted for the difference in the clocks between the two machines). Effectively, this approximates the elapsed time from when the update happened in DMSII to the time when it got applied to the relational database.

AFN/DMS*

The first part of this column is the audit file number of the audit file being processed. If the current DMSII audit file is different than the one being processed both audit file numbers are displayed with a slash between them.

ABSN*

The audit block serial number that the Engine is processing.

Audit Timestamp*

The last timestamp encountered in the audit trail.

Progress*

This column is only meaningful if the Client has fallen behind DMSII by one or more audit files. In this case it indicates approximately how far into the audit file has been processed (expressed as the percentage of blocks processed). If the client is processing the current DMSII audit file this will tend to be close to 100%.

Main Thread Status

This column displays the status of the main thread of the Client program, which is responsible for reading DMSII data. If using multi-threaded updates the DMSII data buffers get put on an update worker's work queue for processing. If not using multi-threaded updates, the main thread will be do all of the processing before reading the next DMSII buffer (this does not lead to optimum resource utilization om a modern servers that has multiple CPUs). If the main thread is waiting on something, this column will show that information.

BCP Thread status

(Window clients only) This column displays the status of the bulk loader thread of the Client program, which is responsible for launching the bulk loader and waiting for it to complete the operation. If the bulk loader thread is waiting on something, this column will show that information.

Index Thread Status

This column displays the access method for reading audit file. For Enterprise Server there are 3 possible values: direct disk, indirect disk and cache. For DBServer on the NCP this will always be host audit.

AF Origin

DMS Recs

This column is the count of DMSII records processed.

DB Ops Count

This column is the count of SQL updates executed. If you do not flatten OCCURS clauses this count can be significantly larger than the DMS recs count.

DB Suppressed Updates

When you use do not flatten an OCCURS clause and enable the **Optimize Updates** parameter redundant updates to secondary tables are suppressed. This is the number of updates that were suppressed because they would not change anything.

DB Filtered Records

When using occurs table filtering, you define filtering conditions that discard secondary table records whose values indicate that the data they contain is meaningless. For example, in an OCCURS 12 TIMES clause it is possible that only the first 3 occurrences are meaningful data. This is the count of the number of records that were filtered out.

DB Discarded Records

This is the count of the records that could not be used to update the relational database, as their key columns had bad data.

DB Error Records

This is a count of the records that had data errors, which caused some columns to be set to NULL.

Bytes Extracted

This is the count of actual DMSII data bytes that was sent to the Client.

Total Bytes Received

This is the count of bytes sent to the Client, it includes DMSII data and the protocol overhead.

BI Bytes Extracted

The client uses BI/AI pairs to process updates in some situations, including processing occurs tables when the **Optimize Updates** parameter is set to True, or when there is an actual key change in a DMSII record, and lastly, when processing data sets that have OCCURS DEPENDING ON clauses. This column is the number of before image bytes sent to the client.

Throughput*

Throughput is calculated by dividing the bytes received by the elapsed time. It is expressed in kilo-bytes per second.

Total throughput

Total throughput is calculated by dividing the total bytes received by the elapsed time.

Records Per Second*

This is rate at which the Client is receiving DMSII records from the Databridge server.

Updates Per Second*

This is rate at which the Client is executing SQL updates.

% Wait for TCP

This represents the percentage of time the main thread was waiting for TCP input from the server.

% Wait for DMS Buffer

This represents the percentage of time the main thread was waiting for a DMSII buffer to become available before it read DMSII data.

% Wait for WS

This represents the percentage of time the main thread was waiting for a working storage block to queue a DMSII buffer on an update thread worker's work queue. These small blocks are pre-allocated, and the Client should have a sufficient amount. DMSII buffers can be placed on multiple worker's queues, possibly multiple times, as a DMSII buffer can generate updates for many tables. To be able to do this, we use working storage blocks, which contain the links for the items on the work queue and point to the DMSII buffer.

% Wait for SQL

This represents the percentage of time the main thread was waiting for a SQL query to finish.

Wait for Transactions

This represents the percentage of time the main thread was waiting for a commit to finish.

CPU Timestamp

This represents the percentage of time the main thread was running.

Stop Run Time

This column represents the time at which the Client will stop as a result of an operator Stop command that specifies the time when the Client will stop.

Stop AFN

This column displays the last AFN that the Client will process before stopping. It is the result of an operator issuing a stop after AFN command.

Last Run Exit Code*

If the data source does not have an active run this column displays the exit code of the last run. Otherwise, the column is blank.

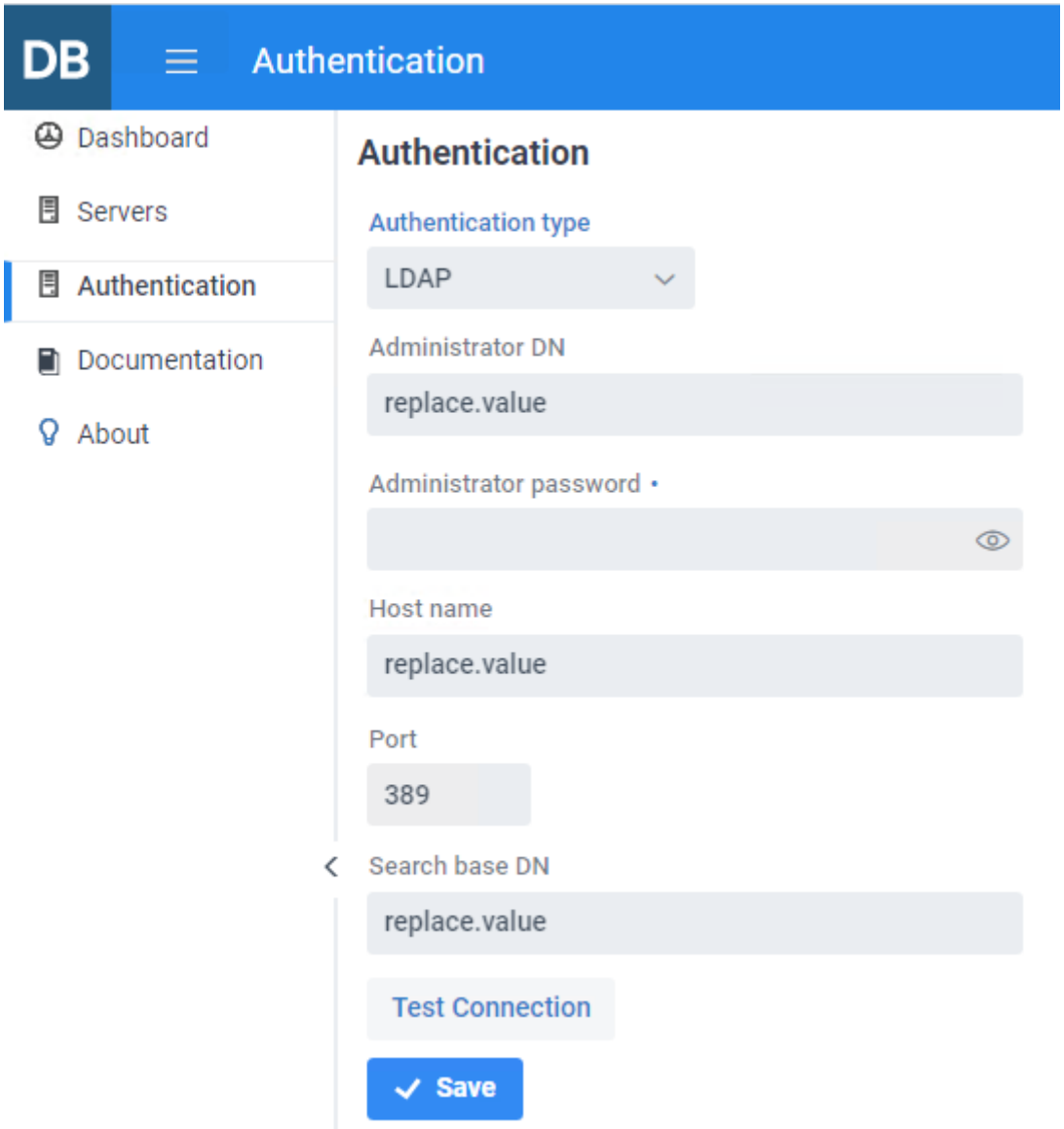
Next Run Time*

If the data source does not have an active run and the service's scheduling will start a run, this column represents the scheduled launch time for the run. Otherwise, the column is blank.

6. Setting up Authentication

The following content is available in the *Databridge Installation Guide*.

The first time the console is run use `dbridge` for the **User Name** and **Password**. Once signed in, select the menu icon ( ) on the left-side of the header, this will expand a left-side navigation menu. Select **Authentication** to configure authentication for the Databridge Administrative Console.



Chose the desired authentication mode using the drop down titled **Authentication type**, the available options are **LDAP** (default) and **Simple authentication**.

6.1 Setting up LDAP

To setup LDAP for the site contact IT personnel and ask them to provide you with the information needed.

Remote desktop to your site's network authentication server and open an elevated Command Prompt (or have someone authorized perform the task below).

1. To search the entire domain use the first line returned from a `dsquery*` command as the base DN. To limit the scope of the base DN, one can use an OU. Running a `dsquery OU` will provide a list of Organizational Units from which the base DN can be selected. Enter this information into the **Search base DN** input field.

   > **Example**
   >
   > "CN=example, CN=com" or "OU=DBUser, CN=example, CN=com"

2. To get the value for the **Administrator DN** input field enter the command `"dsquery user -name` *username*, where *username* is the userid of an administrator on the authentication server. Copy the output line, and paste it into the **Administrator DN** input field.
3. Enter the *hostname* of the network authentication server into the **Host Name** input field.
4. Enter the password for the administrator(s) selected in step 2 into the **Administrator password** input field.
5. Ensure that the default port of 389 is correct for your site.
6. Select **Save**.

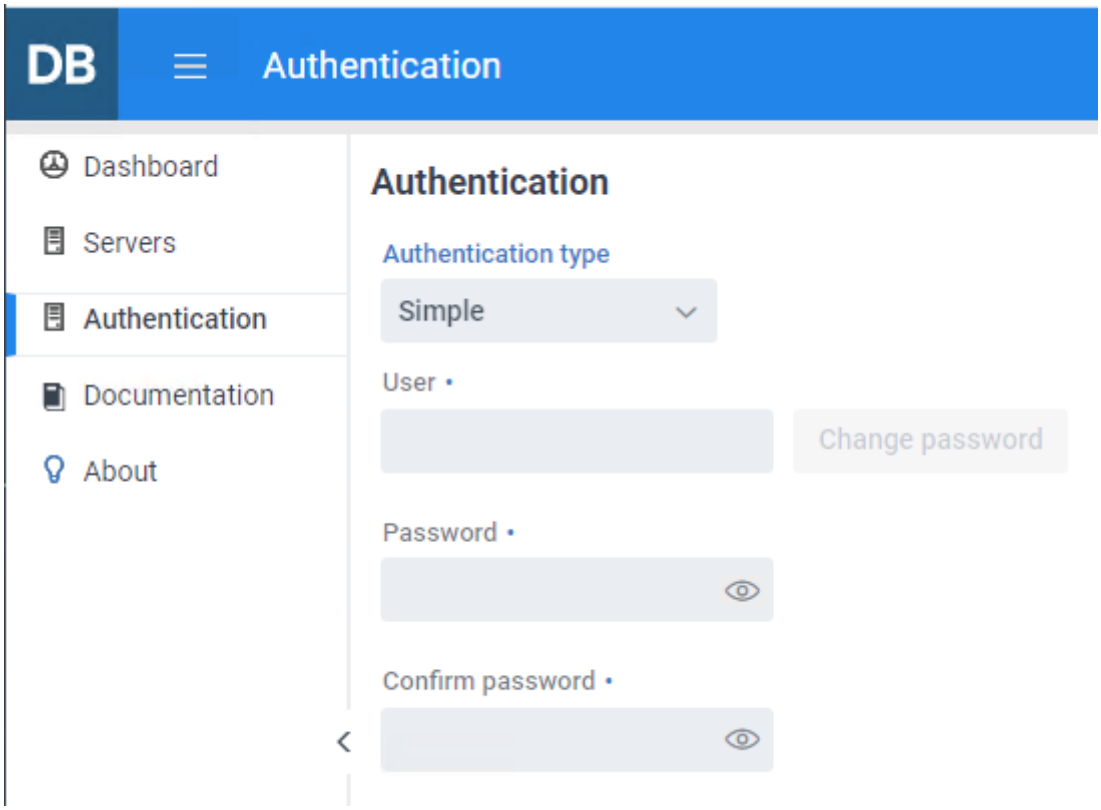
   These changes will only take effect after restarting the service. At this point, you will be able to connect to the Administrative Console from a browser using your network credentials, though this alone is not enough to gain access to the Client Managers.

   For access to the Client Manager(s), userids will need to be given access to a Client Manager. To grant access, the administrator(s) will need to assign users a role to access the **Client Manager(s)**. To do this, assign a predefined role (**Administrator**, **Operator**, **User**, or **Custom**) to the user. For more information on user permissions, see the Manage Users section of this guide.

> **Caution**
>
> **How do you gain access to the Client Manager to assign roles?**
>
> When upgrading, the userids and roles are honored in the new Administrative Console.
> Other ways to gain access:
>
> - `dbridge` will be available from the Client Manager configuration file to get initial access to the Client Manager page with permissions to manage users.
> - the `dbridge` userid and password is automatically added as an administrator for the first sign-on to the Client Manager after an upgrade. The service will automatically detect if there was an upgrade the first time it is started.

6.2 Creating Userid/Password Pairs

When **Simple** authentication is setup, the administrator needs to add the userid/password pairs as seen in this screenshot.



Enter as many userid/password pairs as needed. Users can access this dialog in the same way as an administrator, but they will only be able to view and change their own password. Userids and passwords are stored in the file `osp-users.csv` in the sub-directory **microservices\auth-service** of the install directory for the Administrative Console, and all passwords are encrypted.

Userids assigned with **Simple** authentication behave in the same manner as the LDAP userids. Userids must be registered with each Client Manager in exactly the same way as LDAP users.

For more information on granting access to the Client Manager(s), see the Manage Users section.

> **Caution**
>
> Change the password for the userid `dbridge` as soon as possible. Leaving the password value `dbridge` can lead to unauthorized access to the Administrative Console.

7. Client Manager

7.1 Client Manager

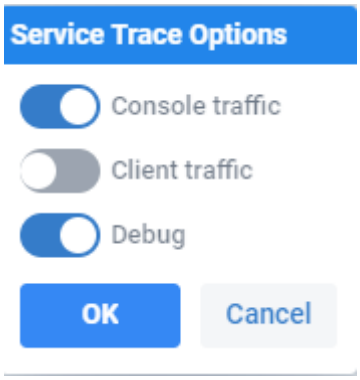
The **Client Managers** are listed along with an **Actions** button.

The following Actions are available from the Client Manager page.

- Service Trace Options: Set or clear tracing and debugging options in the service.
- Switch Service Log: The service will close the current log file and open a new one.
- Manage Users: Add, edit, or remove users for the Administrative Console and set permissions levels for each user.
- Set bconsole Password: Administrators can set the password for a userid that will be used by the bconsole program.

7.2 Service Trace Options

Selecting **Service Trace Options** brings up the following dialog which allows you to set or clear tracing and debugging options in the service. The changes made will only persist until the service is stopped. Changes made in this dialog are note saved to the configuration file. It is recommended to keep these options disabled unless directed to do so by Micro Focus Customer Support, as running the service with debugging and tracing enabled will generate large log files. This dialog should be utilized as a diagnostics tool for collecting information when a problem is encountered.



- **Console traffic:** controls the tracing of the RPC traffic between the various Administrative Console sessions and the Client Manager service.
- **Client traffic:** controls the tracing of the RPC traffic between the launched clients and the Client Manager service. A lot of this traffic will also appear in the **Console traffic** so you almost never would want to enable both of the trace options at these same time.
- **Debug:** enables some additional logging information, which includes the RPC type to make a trace file easier to read.

> **Tip**
>
> All three options should remain disabled during normal operation.

7.3 Switch Service Logging

Clicking on **Switch Service Log** causes the service to close the current log file and open a new one.

The file names include the date when the file is created (for example `cp20210722.log`). If the old log file was created on the same day as the new file, the new file name will also contain the current time to make the name unique (for example `cp20210722_144825.log`). When the service switches to using a new log file the last line of the old log file will contain a line with the text:

```
14:48:25 Log file switched to "logs\cp20210722_144825.log" (Operator Keyin)
```

A confirmation message will appear in the console following the successful switch.

7.4 Manage Users

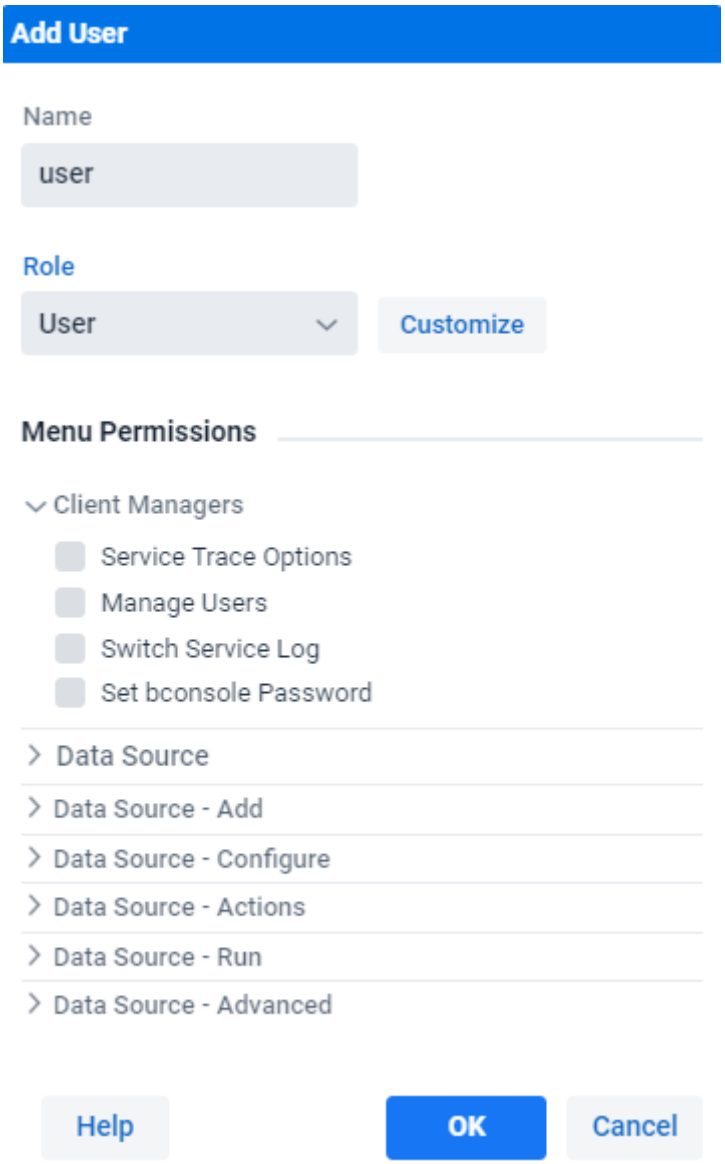
Use this section to add, edit, or remove users from the Administrative Console and set permissions.

> **Note**
>
> The Authentication of browser users does not guarantee access to a particular Client Manager service.

Before a user can successfully connect to a Client Manager, the userid must be added to the service's configuration file. This page allows an administrator to grant permissions on a per user basis. Permissions can be set as pre-defined roles (such as Administrator), or, can be customized to include access to specific menu commands in the Administrative Console.

7.4.1 To add a user ID and assign permissions

1. From the **Client Manager** page, select **Actions** > **Manage Users**.
2. On the **Manage Service User IDs** page, select **+Add** and enter the user ID in the **Add User** input field as seen below.



3. From the **Role** list-box, select the role that best fits the level of usage.
   - **Administrator:** Has full privileges to all menu items. This includes configuration, customizing data sources, and starting/stopping client runs.
   - **Operator:** Can manually start and stop client runs.
   - **User:** Can view, but not modify properties and operations in the Administrative Console.
4. If you want to customize the user ID's permissions, select **Customize** and continue the following steps. If you're done, select **OK**.
5. When **Customize** is selected, customize the permissions in the **Menu Permissions** section.

   The menu commands are arranged in groups that correspond to the Administrative Console menus in the user interface.
6. Select **OK** to close the **Manage Service User IDs** dialog box.

> **Note**
>
> You cannot limit access to the Dashboard, Statistics or Documentation menu items as these items are accessible to all users.

> **Tip**
>
> After you've created a customized user ID, you can copy its permissions and assign them to a new user ID. This is useful when you need to create multiple user IDs with the same permissions.

7.4.2 To copy permissions to a new user

1. From the **Manage Service User IDs** page find the user permissions that will be copied and select the **Copy** icon  associated to that user.
2. By selecting the **Copy** icon  , an **Add User** pane will load with the copied permissions.

   The copied permissions appear as **Custom** in the Roles menu and the username will display as **Copy of...**.
3. Enter a username for the new user and select **OK**.

7.4.3 To remove a user

1. Select the specific user from the **User List**.
2. Do one of the following:

   Select **Settings** > **Delete**

   -or-

   Select the corresponding **Delete** icon  , and select **Yes** when prompted to delete the specific user.

> **Note**
>
> You cannot delete the default user ID or any currently active user IDs.

7.5 Set and Change the Console Password

Use this option to change the password for a batch console user ID. For Administrative Console users, the authentication occurs between the browser and the Administrative Console server where the browser user needs to first be authenticated using either **LDAP** or **Simple Authentication**. Administrative Console users do not need a password to get granted access to the Client Manager service. However, batch console users still use the same authentication method as the 6.6 console users. If you are using the **bconsole** utility in scripts that are not launched by the service, you need to supply a password along with the userid to gain access to the Client Manager service.

This option allows the Administrative Console administrator to set the password for a userid that will be used by the bconsole program. The same userid can be used by an Administrative Console and bconsole users, as the Client Manager detects the type of console connection that was established.

The service manages user IDs and passwords in the service configuration file (`dbcontrol.cfg`). By updating the service configuration file using the Administrative Console, the administrator can allow a batch console user access to the service. Before setting the **bconsole** password, you must first create the batch console userid. If the same userid is used by an Administrative Console and **bconsole** user, this will have no effect on the **bconsole** user, as the permission bits are meaningless in the batch console, which has no UI. The **bconsole** utility is typically used to make the service launch Client runs.

To change the password

- In the **Client Manager** page, select **Actions** > **Set bconsole Password**.

  Complete the **New Password** and **Verify Password** and select **OK**.

The password is encoded and saved in the service configuration file (`dbcontrol.cfg`).

7.6 Client Manager Information (Read-Only Properties)

From the **Client Managers Properties** page, basic information about the selected server can be viewed.

> **Important**
>
> This page only displays read-only properties for the selected Client Manager.

7.6.1 General

The **Servers Information** page shows the read-only properties of the Server selected from the Servers list.

- **Address:** displays the **Host name** or **IP Address**.
- **Protocol:** displays the protocol number.
- **Port:** displays the port number for the Server.

8. Working with Data Sources

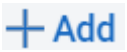
8.1 Working with Datasources

The Administrative Console data source page allows the user to view and manage data sources. You can accomplish each phase of the replication process by using menu commands and options. See Data Sources Commands and Options below for instructions on how to use the data sources page.

8.1.1 Data Source Commands and Options

Select a data source button below to see the available commands and options. Links to the specific documentation will be available through the tabbed options below.

Select the buttons below for more information



- New
- Existing

- Overview
- Configure
- Customize
- Data Sets
- Data Sets State Info
- Information
- Properties

Run

- Overview
- Stop
- Commit Parameters
- Switch Trace
- Client Statistics
- Abort
- Switch Client Log

Actions

- Overview
- Process
- Generate Scripts
- Reorganize
- Define/Redefine

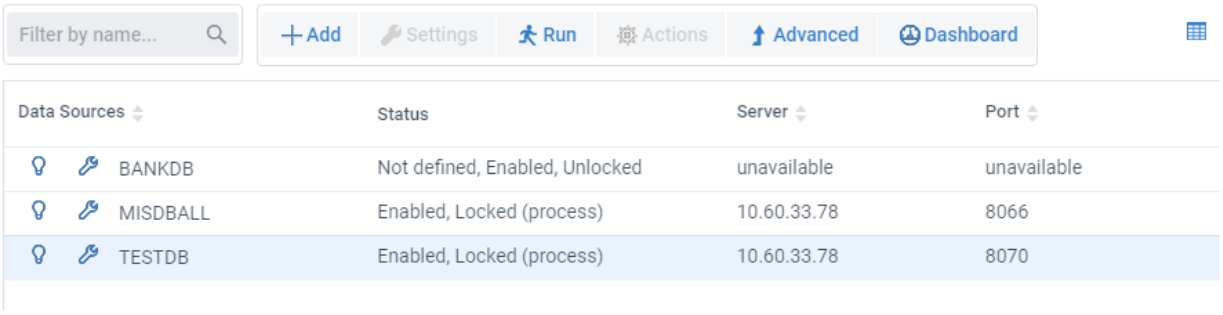
Advanced

- Overview
- Trace and Log Options
- Change Database Password
- Change SOURCE Password
- Export Client Configuration
- Unload, Reload, Remove Data Source
- Process (with options)
- Clone Data Set
- Log Control Tables
- Redefine (with options)
- Generate All Scripts
- Refresh Data Sets
- Reorg (Multi-source, 2nd source)
- Create and Run User Scripts
- Run History

- See Dashboard walkthrough

8.1.2 Data Sources Table

The data sources view for a Client Manager is a table that has a row for each data source included in the service's configuration file.



The first column is the name of data source. To the left of it are the two icons **Properties** and **Information** in the **Settings** menu.

The **Status** column contains comma separated status values, which include:

- **Not defined**

  This status indicates that a **Define** or **Customize** command has not yet been run for the data source. You will see this status immediately after you add a new data source or after you remove a data source without removing the working directory.

- **Enabled**

  This status indicates that the data source is not disabled. You should see this for all data sources under normal circumstances.

- **Disabled**

  This status indicates that the data source is disabled. A data source is disabled by the service when a `process` or `clone` command terminates with an exit status that is not subject to automatic retry attempts (that is, a fatal error), or when the retry attempts for recoverable exit statuses reaches the maximum configured value (3 by default). This implies that the Administrative Console operator will not be able to start a **Process** or **Clone** command and that the service's scheduling will not attempt to start runs until the data source is enabled by the Administrative Console operator. This is a safeguard to prevent endless Client runs that perpetually fail.

- **Unlocked**

  Most Client commands lock the data source while they are running, as you cannot have two `process` command runs updating the database at the same time. The service and the Administrative Console keep track of the state of the data sources and prevent runs from being started when the data source is locked. The Client uses its own locking mechanism which is implemented as a file that is opened with exclusive access, so that if another command tries to lock the data source it gets a "file in use" error, which terminates the run. The Administrative Console protects against this, except in the case where you start a command using the command-line client. For this reason, you should never mix the two modes of operations.

- **Locked** (*operation*)

  This status indicates that the data source is locked, *operation* is the name of the command being executed. This includes `process`, `clone`, and `configure` (indicates that a **Customize** command is running). When this status is displayed most of the menu items found in the menu bar will be disabled until the run ends.

- **Needs generate**

  This status indicates that one or more of the scripts in the data source's **dbscripts** directory are no longer up-to-date, and that you need to run a **Generate Scripts** command to remedy this situation.

- **Needs redefine**

  This status indicates that a change in the data sources settings requires that a **Redefine** command be run to update the Client control tables, which are no longer in sync with the data source's global parameters.

- **Needs reorg**

  Following a **Redefine** or **Customize** command, if a layout change is detected in one or more tables the command creates temporary scripts in the data source's working directory to alter the tables so their layouts reflect the changes. The **Reorganize** command in this example can be seen as the second phase of a **Redefine** command.

  The **Needs reorg** status indicates that you need to run a **Reorganize** command. It is recommended that you examine the scripts before running the **Reorganize** command, as some of the SQL commands may take a long time to execute on large tables.

  If this is the case, enabline the `internal_clone` option (using the **Configure** option from the **Settings** drop-down) and rerunning the **Customize** or the **Redefine** command will yield better results. The internal clone renames the table and uses a select into SQL statement to copy the data from the renamed table to a new copy of the table while providing values for added columns. When completed, it deletes the renamed original table. This command runs at bulk-loader speed (hence the name internal clone). If this does not look like it will work because of disk space limitations (disk space to a hold second copy of the table during the internal clone), your only remaining option would be to re-clone the data set.

  For more information on the **Configure** option in the **Settings** drop-down see the Global Parameters page.

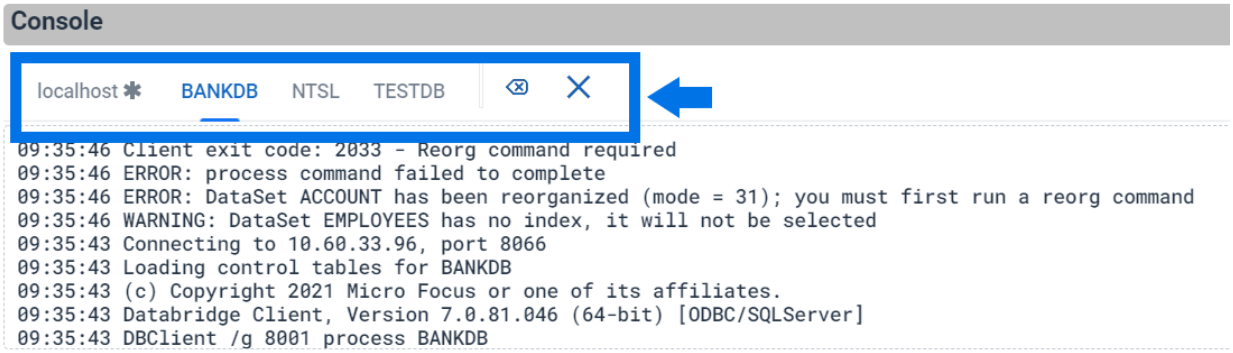
- **Reorganized**

  This status indicates that a **Process** command encountered a DMSII reorganization and exited with an exit code of 2. When the **Customize** command is run in this situation it is limited to updating the data sets, DMS items, data tables, and data items that were affected by the DMSII reorganization unless the filtering was edited to show all data sets.

The next two columns are the domain name (or the IP address) and the port number for the Databridge Server.
This server can be either DBServer on an MCP system or Enterprise Server on a Windows machines.

8.1.3 Console Output

The Console pane at the bottom of the data sources page displays Client-related activity for each data source, including log output, exit codes, and commands. (This information is the equivalent of the program output displayed when running the command-line Client.)



Note

Each data source has its own console tab that can be accessed by selecting the data source tab in the console pane. As seen in the screenshot above, **BANKDB** console output is currently selected. Users can select any data source and view the output in this pane. If you select a data source in the data sources page its corresponding tab in the console pane will be automatically selected.

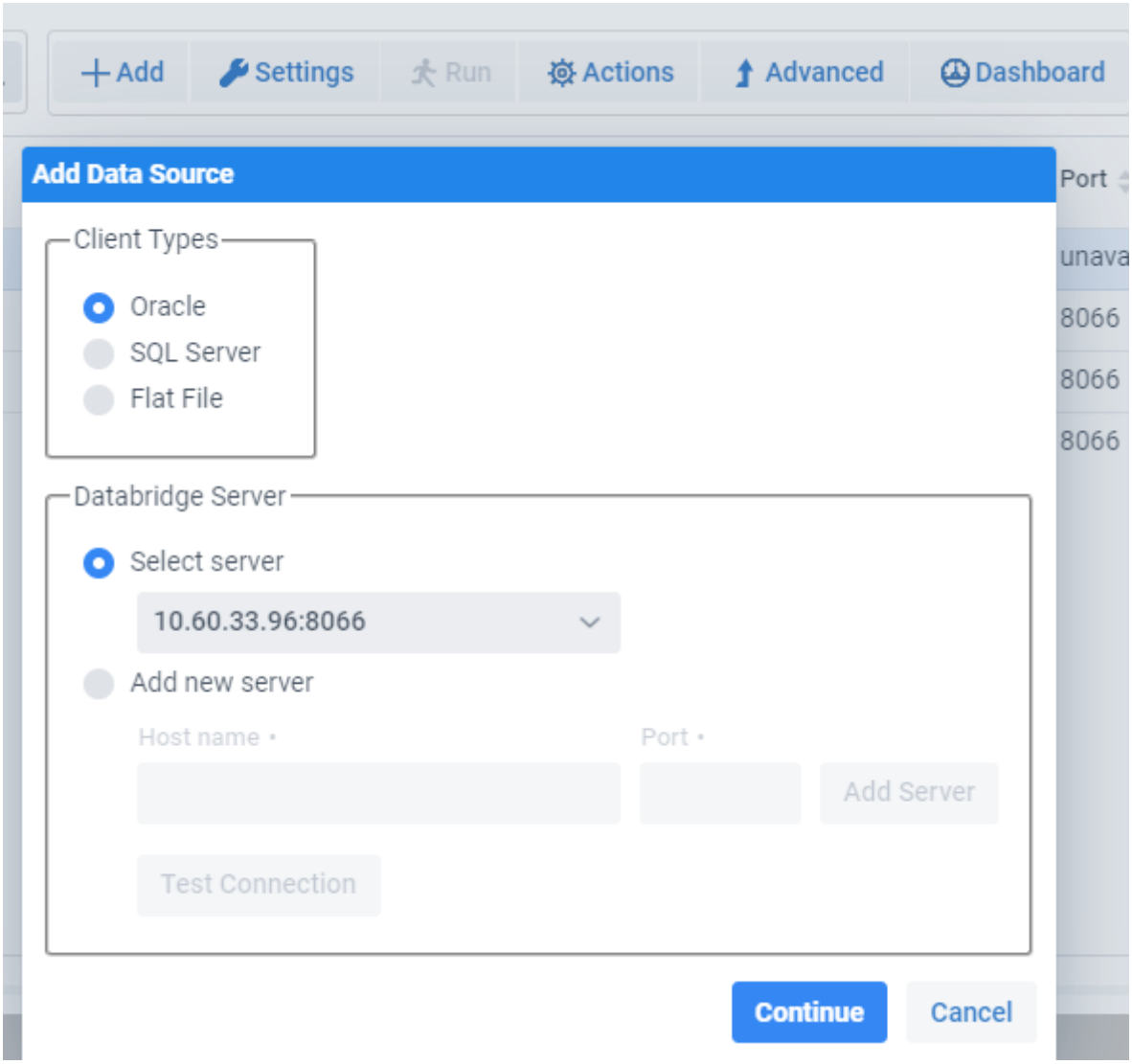
8.2 Add

8.2.1 Add New Data Source Dialog

Use this procedure to specify data sources for replication. You can add multiple data sources to a service. Ensure that the Databridge Server is running and the relational database is configured in accordance with the manufacturer's instructions. SQL Server databases must have an ODBC data source created using **Control Panel\All Control Panel Items\Administrative Tools>ODBC Data Sources(64-bit)** before you can continue.

Adding a New Data Source

1. Navigate to the Client Manager's data source page by selecting **Databridge Server** > **Client Managers** > *clientmanagername*, or, by clicking on the *clientmanagername* link on the Dashboard, where *clientmanagername* is the name or IP address of the Client Manager to which the data will be added.
2. From the data source page select the **+Add** button in the menu bar and select **New** from the drop-down options.



The Client Type group will only appear if you have more than one type of Client installed (e.g. SQL Server and Flat File). If there are multiple Client installs, select the desired install as it defaults to the first one in the list unless specified.

> **Note**
>
> The Client Manager service does not currently support the Kafka Client.

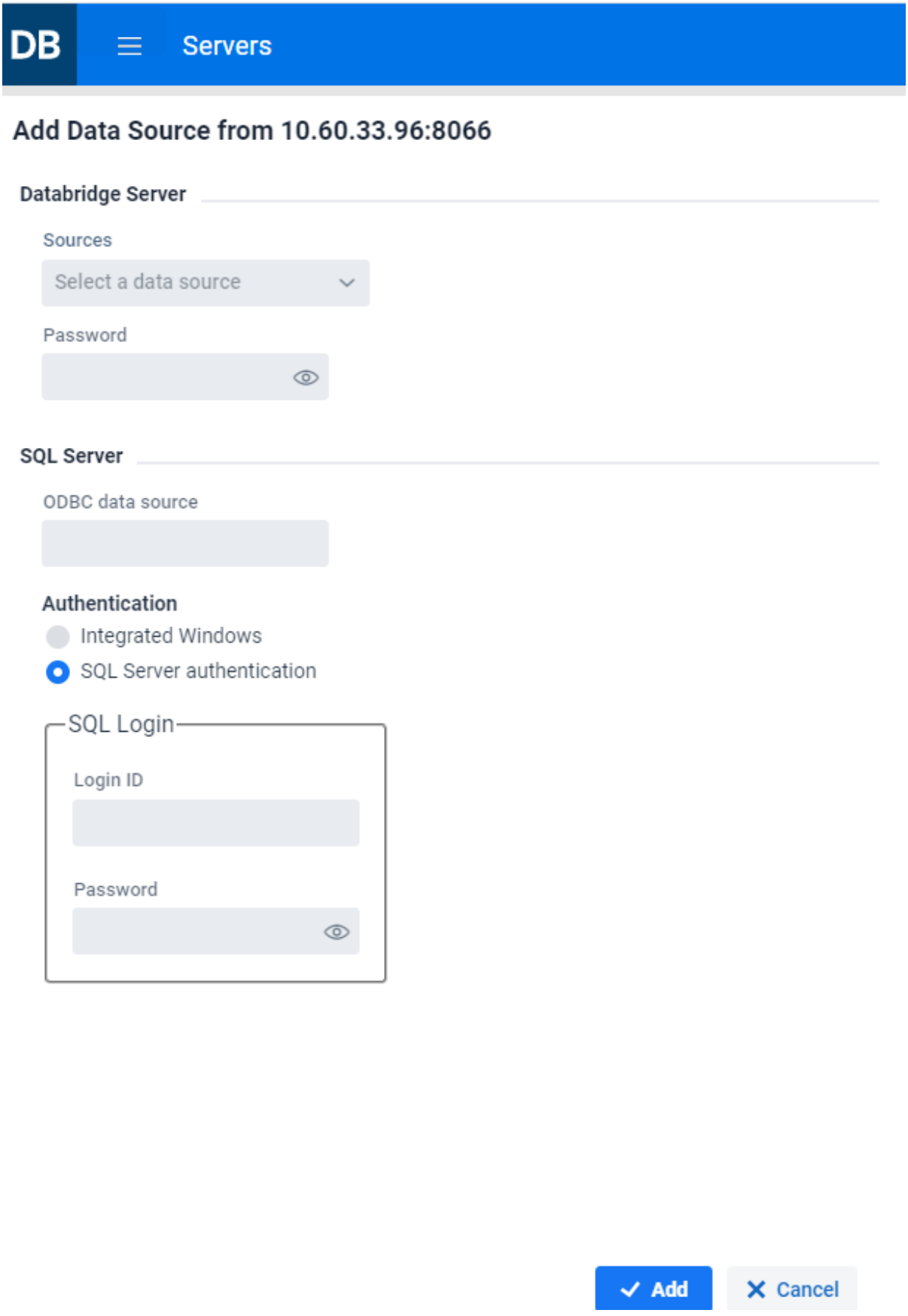
3. Select a server from the drop-down list provided under the **Select Server** option.

   If the desired server is not in the list, select **Add new server**. When **Add new server** is enabled, enter the domain name or IP address of the Databridge server (this can be either a DBServer on

an MCP system or on Enterprise Server) in the **Host name** edit box along with the corresponding port number. Once this information has been added, select the **Add Server** button. Once you have selected a server using one of these two options, select the **Continue** button to move to the next page of the dialog.

4. Select the desired data source from the **Sources** drop-down.

   The drop-down list is generated from the specified server and removes the data sources that are already configured to prevent you from adding a data source that has already been configured in the service.



Tip

To find a specific data source type the first letter in the drop-down field to filter results.

5. Do one of the following:

   - For SQL Server, in the **ODBC data source** input field, type the name of the ODBC data source you created.
   - For Oracle, type the database name.

6. Specify the type of authentication:

   - For SQL Server, select **Integrated Windows authentication** or **SQL Server authentication**. If you select SQL Server authentication, type the SQL login ID and password.
   - If you are using the Oracle instant client to connect to a remote Oracle database, the database name will be from "*node*/*database*", where *node* is the IP address or the domain name of the machine where the Oracle database resides.

7. Click **Add**.

> **Note**
>
> *If the data source parameters are valid*, the data source appears in the data source view of the Client Manager. The service adds the specified login ID and password to the Client configuration file (`dbridge.cfg`).
>
> *If the data source parameters are invalid*, an error will be displayed in a pop-up message. Correct your entries and try again by selecting **Add**.

8.2.2 Add an Existing Data Source

To add an existing data source, the working directory must be present as a subdirectory of the service's working directory. The working directory will need to be moved or copied to a subdirectory of the service's working directory to successfully add an existing data source.

When adding an existing data source, the Administrative Console issues an RPC to get the list of subdirectories in the service's working directory that meet the requirements for being working directories for a data source. Data sources that are already included in the service's configuration file will be removed from the list.

Add an Existing Data Source

1. From the Data Sources page select the **Add** button.

   From the **Add** button dropdown select **Existing**
2. On the **Add Existing Data Source** page, use the **Select a data source** field to select the desired data source form the dropdown options.

   > **Note**
   >
   > To quickly find a data source, type the first letter of the desired data source into the **Select a data source** field.
3. Select the **Save** button at the bottom right of the page to add an existing data source.

8.3 Settings

8.3.1 Data Source Settings

From the settings dropdown on the Data Sources page you can access the following options in the dropdown.

- Configure: The Configure page allows administratorss to customize the global parameters at the data source level.
- Customize: Use the **Customize** option to map individual DMSII data sets (and their items) and customize the table layout of the relational database.
- Data Sets: From the **Data Sources** page you can access a list of data sets for any given data source.
- Data Sets State Info: Use the **Data Sets State Info** page to modify the state information for all data sets that are in **Tracking** mode.
- Information: View the read-only data source properties for the selected data source.
- Properties: View and configure and editable properties for the selected data source.

8.3.2 Settings > Configure

Getting there

The **Configure** page allows administrators to customize the global parameters at the data source level with the following options:

- Bulk Loader
- Customizing - General
  - Advanced
  - History Tables
  - SQL Data Types
  - SQL Suffixes
  - Translations
  - User Columns Section One
  - User Columns Section Two
  - User Scripts
- Logging
  - Service Log
  - Trace Log
- Processing
  - Advanced
  - Date and Time
  - Engine and Enterprise Server
  - DMSII Data Error Handling
  - Error Recovery
  - Schedule
  - Statistics
  - Stop Conditions
- PCSpan
  - Advanced
- Encryption

The pages linked above can be found in the Global Parameters section in the Table of Contents.

8.3.3 Customizing Data Sets

Use this page to map individual DMSII data sets (and their items) and customize the table layout of the relational database.

Customizations should be made before cloning a data source to prevent having to run a re-clone. To make customizations that affect the entire data source, use the **Configure** option available from the **Settings** drop-down on the data sources page. See Customize the Client Configuration for more information.

> **Note**
>
> The **customize** command is a cooperative process between the **DBCIntCfgServer** program, the Client Manager service, the code in the Administrative Console server, and the browser. You should never use the browser back button during this command ,or, close the browser before you exit the **Customize** command. Use the **Done** buttons or the links provided in the Administrative Console to complete this command. Closing the browser when making customizations can result in the **DBCIntCfgServer** program being left in a bad state. When this happens, you can wait for the timeout, or end the task by using task manager.

When **Customize** is selected the data set view will be displayed. From this view, customizations can be made at the data set level. Some examples of customizations that can be made can be seen below:

- Deselecting specific data sets to not be replicated.
- Setting data set specific options that override the configured global parameters specified in the Client configuration file.
- Link virtual data sets and the real data sets from which they are derived.

DMS Items and Relational Data Item and Table Customizations

From the data sets view you can select a data set to get to its DMS item view where you can perform customizations on the data set's DMS items. The following customizations can be made from the DMS Items Properties page.

- Modify the way an item is replicated.

  For example, a date represented as a NUMBER(8) in DMSII can be replicated as a relational database date.
- Modify an item with an OCCURS clause so it can be mapped to a secondary table or be flattened within the table.

From the **DMS Item Properties** page select the **Relational** tab to access the relational view where you can customize data tables and data items. The following customizations can be made from the Data Item and Data Table Properties page:

- Rename data tables and data items
- Change the order of columns in a table
- Modify the data type of an item, or making its length bigger.
- (SQL Server Client only) Apply data masking to sensitive items

The **Done** button in both the **DMS Items** and **Relational** tabs will return to the data sets view.

To exit the **Customize** command return to the data set view and select the **Done** button.

8.3.4 Manage Data Sets

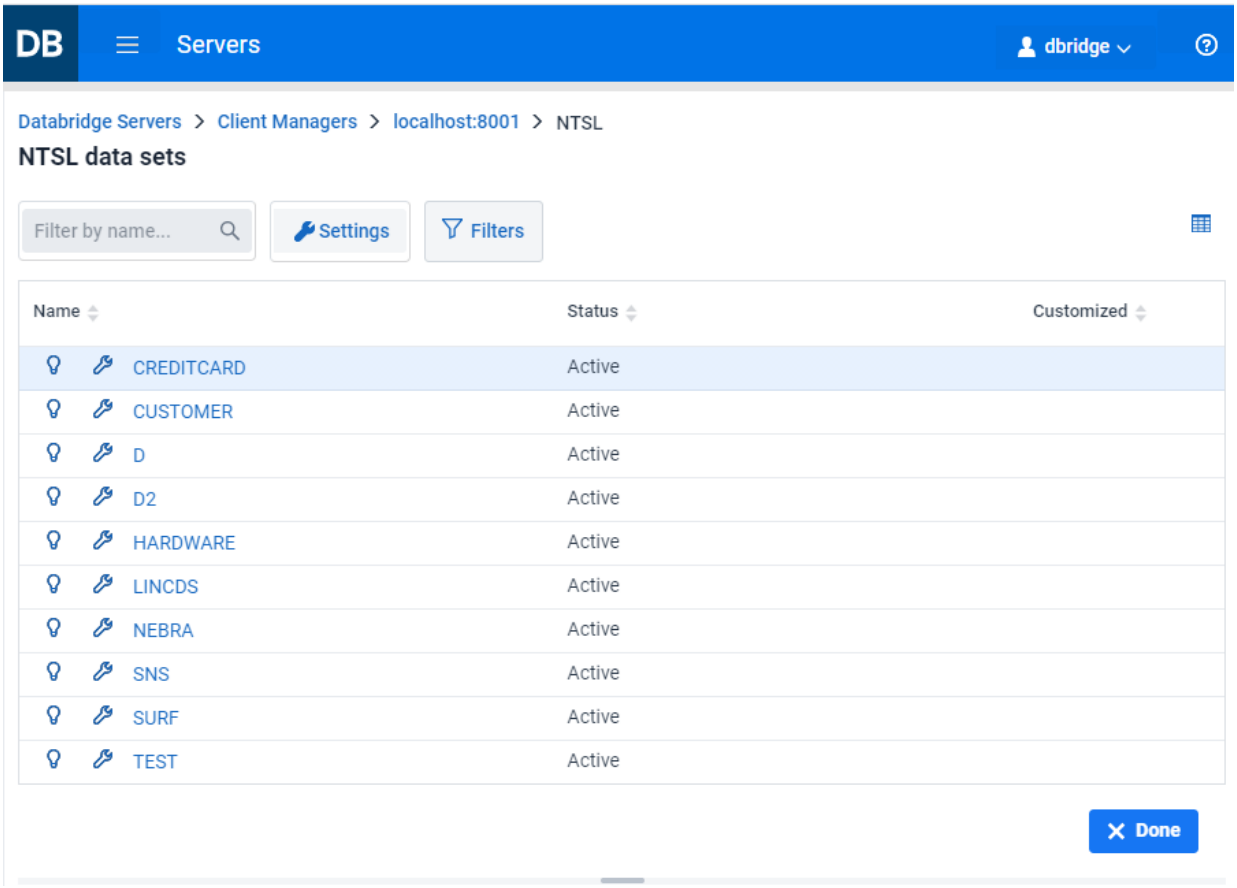
Getting There

From the **Data Sources** page you can access a list of data sets for any given data source by selecting the **Data Sets** option from the 

dropdown.

Tour the Manage data sets page

See below for a description of the tabe options.



Filter by name...	Enter a string of characters or the name of a specific data set to sort the list of available data sets. This input field can be used to avoid scrolling through a long list of data sets.
Settings	By selecting the **Settings** dropdown you can access the **Information** or **Properties** for any specific data set that is selected in the list view.
Filters	Select the filters (**Active**, **Inactive**, and **Reorganized**) to display the desired data sets in the list view of the **Manage data sets** page.
	Select this icon to configure which columns will be displayed in the data sets list view.

View a specific data sets Properties page

1. From the **Manage data sets** page, select the specific data set so it is highlighted like the **CREDITCARD** data set in the above screenshot.
2. Do one of the following:
   - Select the icon to the left of the selected data set to open the **Data Set Properties** page.

-or-

- From the 

Settings button drop-down select **Properties**.

View a specific data set Information page

1. From the **Manage data sets** page, select the specific data set so it is highlighted like the **CREDITCARD** data set in the above screenshot.
2. Do one of the following:
   - Select the Lightbulb  icon to the left of the selected data set to open the **Data Set Information** page.

-or-

- From the 

Settings button drop-down select **Information**.

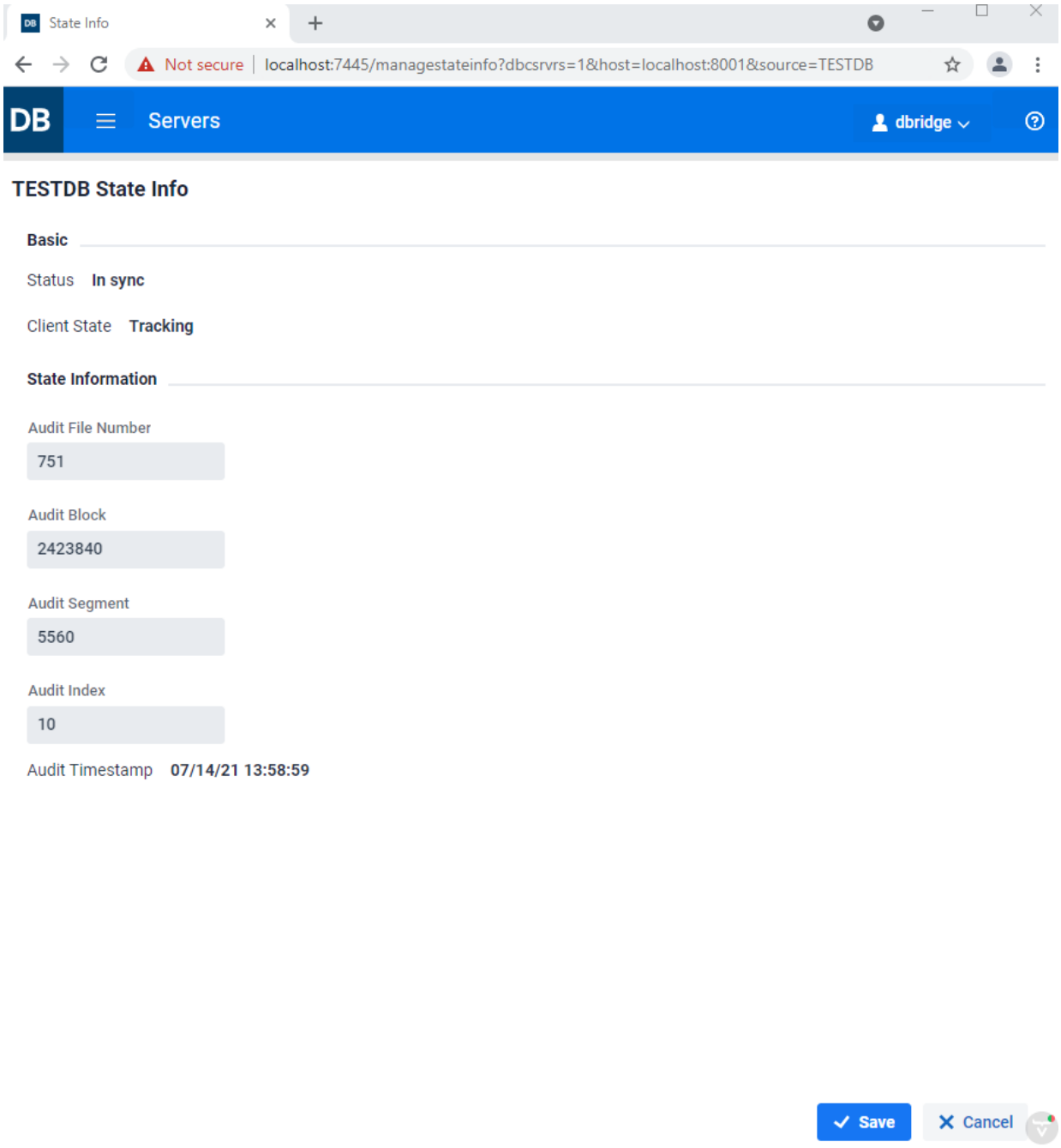
8.3.5 Data Sets State Info

From the **Data Sources** page you can modify the state information for all data sets whose **Active** property is True, have a **Mode** property of 2 (Tracking). This dialog is used to skip over a section of the audit trail when the Databridge Engine is unable to process a block in the audit trail.

> **Caution**
>
> If you get into a situation where the Databridge Engine is unable to process an audit file call Micro Focus Customer Support before attempting to correct the situation. You need to be very careful as you risk corrupting the relational database. You should probably revoke the permission to execute this command for all users using the **Manage Users** command. You can always change this if you run into a situation where it is needed.

To run this command, select the **Data Sets State Info** command from the **Settings** 

drop-down button. This will open the following dialog:



To find the quiet point, run the **DBInfo** utility that is provided as part of the Databridge software on the MCP system. Refer to the *Databridge Host Administrator's Guide* for documentation on how to run **DBInfo**. Enter the values in the corresponding input fields on the page and select the **Save** button in the lower right corner of the page. The audit timestamp will be set to zero, which causes the Databridge Engine not to verify it.

Basic

This section has two read-only properties that describe the state of the Client.

Status

This line will normally have a value of `In sync` that indicates that all the datasets in question have the same state information values. If this is not the case it will display `Multiple values` instead.

Client State

This line will normally have a value of `Tracking`, which indicates that all data sets whose **Active** property is True, have a **Mode** of 2 (Tracking). Other possible values are `Fixup`, `Clone` and `Off line`. A Client state value of `Fixup` indicates that some data sets whose **Active** property is True, have a **Mode** of 1 (Fixup), and none have a **Mode** of 0 (Clone). A Client state value of `Clone` indicates that some data sets whose **Active** property is True, have a **Mode** of 0. Finally, a Client state value of `off line` indicates that there no data sets whose **Mode** is either 0, 1 or 2, which means that the Client is not in a state where it can process updates (for example all data sets could have a mode of 31, which indicates that a **Reorganize** command needs to be run).

State Information

These four edit boxes represent the parts of the State Information that you need to enter, **Audit File Number** (AFN), **Audit Block** (ABSN), **Audit Segment** (segment in the audit file), and **Audit Index** (index in the segment).

8.3.6 Data Source Information (Read-Only Properties)

Getting there

- **Databridge Servers** > **Client Managers** > **Settings** > **Information**

  -or-

  From the **Data Sources** page select the lightbulb icon  for the desired data source.

The read-only properties for a data set appear and are organized in the following sections:

- Basic
- Client
- DMSII Info
- Server
- Version

The properties that can be configured will appear in the Properties page for the data source.

Basic Properties

NAME

The name of the data source.

ENABLED

This property, which can be edited in the **Properties** page for the data source, indicates whether the data source is enabled or not.

AUDIT FILE ORIGIN

This property, which is only meaningful when using Enterprise Server, specifies the origin of the current audit file being processed. The following values are defined for this column:

- **0:** Audit file processed by Databridge Engine
- **1:** Reserved
- **2:** Audit file processed by Enterprise Server using Databridge Engine to access regions. This is referred to as indirect disk.
- **3:** Audit file processed by Enterprise Server using direct disk I/O. This is referred to as direct disk and is the most efficient way to process audit files in terms of host resource utilization.
- **4:** Cached audit file processed by Enterprise Server

Audit File Access Method

This property, specifies which RPC the Client is using to request updates from the Databridge Server. The two possible values are **DBREAD** and **DBWAIT**. The **DBREAD** RPC causes the Databridge Engine to stop the Client by returning a status of `Audit Unavailable` when it reaches the end of the current DMSII audit file. The **DBWAIT** RPC causes the Databridge Engine to enter a wait-and-retry loop when it reaches the end of the current DMSII audit file.

Client Properties

CLIENT TYPE

This property identifies the Client type. The defined values are **SQL Server**, **Oracle** and **Flat File**.

SOURCE ID

This property, which is editable in the **Properties** page for the data source, is the value of the `data_source_id` column in the DATASOURCES Client control table.

TABLE NAM PREFIX

This property, which is editable in the **Properties** page for the data source, is the value of the `tab_name_prefix` column in the DATASOURCES Client control table.

LAST RUN STATUS

This property is the value of the `last_run_status` column in the DATASOURCES client control table, which represents the exit code of the last run. If there is an active run this column has a value of `9999`.

NEXT SCHEDULED RUN

This property is blank if there is an active run, otherwise it will have the time at which the next scheduled **Process** command will be launched by the service, when scheduling is enabled.

ODBC SOURCE NAME

This property is the ODBC data source name that was supplied when the data source was created.

DMSII Info

FILEXTRACT

This property indicates whether the data source is a DMSII database or a FileXtract.

INDEPENDENTTRANS

This property is INDEPENDENTTRAN setting of the DMSII database.

UPDATE LEVEL

The property is the update level of the DMSII database.

Servers

HOST

This property, which is editable in the **Properties** page for the data source, is the value of the `hostname` column in the DATASOURCES Client control table.

PORT

This property, which is editable in the **Properties** page for the data source, is the value of the `hostport` column in the DATASOURCES Client control table.

ENCRYPTED

This property indicates whether the connection to the Databridge server is encrypted or not. Databridge 7.0 only support TLS encryption for connection to **DBServer** on the mainframe.

SERVER

This property specifies the type of the server, the two possible values are **DBServer** and **DBEnterprise**.

Version

This groups contain the versions of the various programs in the Client machine and the Databridge Server. These include:

- **DBServer** The Databridge server on the mainframe.
- **DBEngine** The Databridge Engine.
- **DBSupport** The support library on the mainframe.
- **DBEnterprise** Enterprise Server
- **DBClient** The Databridge Client that runs **Process** and **Clone** commands.
- **DBClntCfgServer** The Databridge Client that handles all Administrative Console commands, except **Process** and **Clone** commands.
- **Service** The Client Manager service (or daemon).

8.3.7 Data Source Properties Page

Getting There

From the **data sources** page, select a data source select **Properties** from the **Settings** button drop-down after selecting the data source in the data sources view, or by selecting the Properties icon  next to the desired data source.

The customizable properties for a data source are organized in the following sections:

- Basic
- Client
- Server

Basic Properties

Properties in this group can only be **Enabled** or **Disabled**. Use the slider to enable or disable the data source. When the data source is disabled the service will not launch `process` and `clone` commands for disabled data sources. A data source is disabled when the Client receives an unrecoverable error, or, after the retry count for error recovery reaches it maximum value (normally 3).

Client Properties

There are two properties in this group, **Source ID** and **Table Name prefix**.

SOURCE ID

This property is the value of the `data_source_id` column in the DATASOURCES Client control table. This property allows you to provide a numeric identifier to distinguish records that belong to a particular data source from other records in a multi-source environment. This value is used to populate the user columns **Data Source ID** or **Data Source ID key** which need to be added to all data sets. See User Columns.

TABLE NAME PREFIX

This property is the value of the `tab_name_prefix` column in the DATASOURCES Client control table. It holds an optional one- to eight-character prefix which is added to all table names in the data source. This prefix, which must be entered, allows you to distinguish between identically named data sets in multiple data sources, without having the **Define**, **Redefine** and **Customize** commands rename tables to eliminate name conflicts. The configuration file parameter `use_column_prefixes` extends this prefix to all column names.`

Server Properties

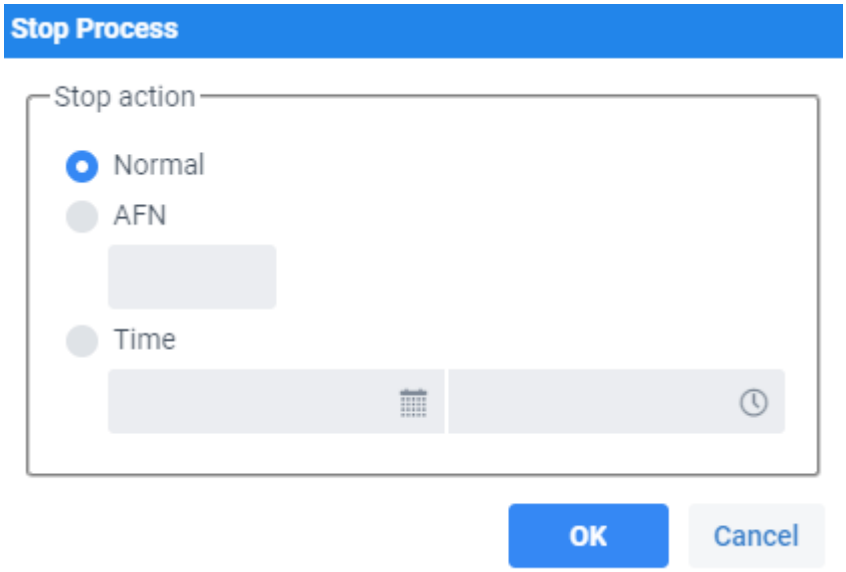
The two properties in this group are the **Host** domain name or IP address and the host **Port**. Change these properties to switch between using **Enterprise Server** and **DBServer** on the MCP, when the **Enterprise Server** data source is identical to the one on **DBServer**. These values are initially set in the **Add > New** command in the data sources view for a Client Manager.

8.4 Data Source Run Commands

The **Run** menu is only enabled when there is an active **DBClient** run for the data source. This program only executes `process` and `clone` commands. See the available options below:

8.4.1 Stop

The **Stop** command allows the operator to stop the Client. When this command/option is selected the following dialog is opened:



See below for the available options:

- **Normal:** indicates that the run will stop at the next quiet point. If you issue this command during the data extraction the run will stop only when the fixup phase begins.
- **AFN:** the run will stop at the first quiet point in the audit file immediately following the one whose number is entered in the provided edit-box has been processed.
- **Time:** Allows the operator to provide a date and time after which the run will stop at the next quiet point.

8.4.2 Abort

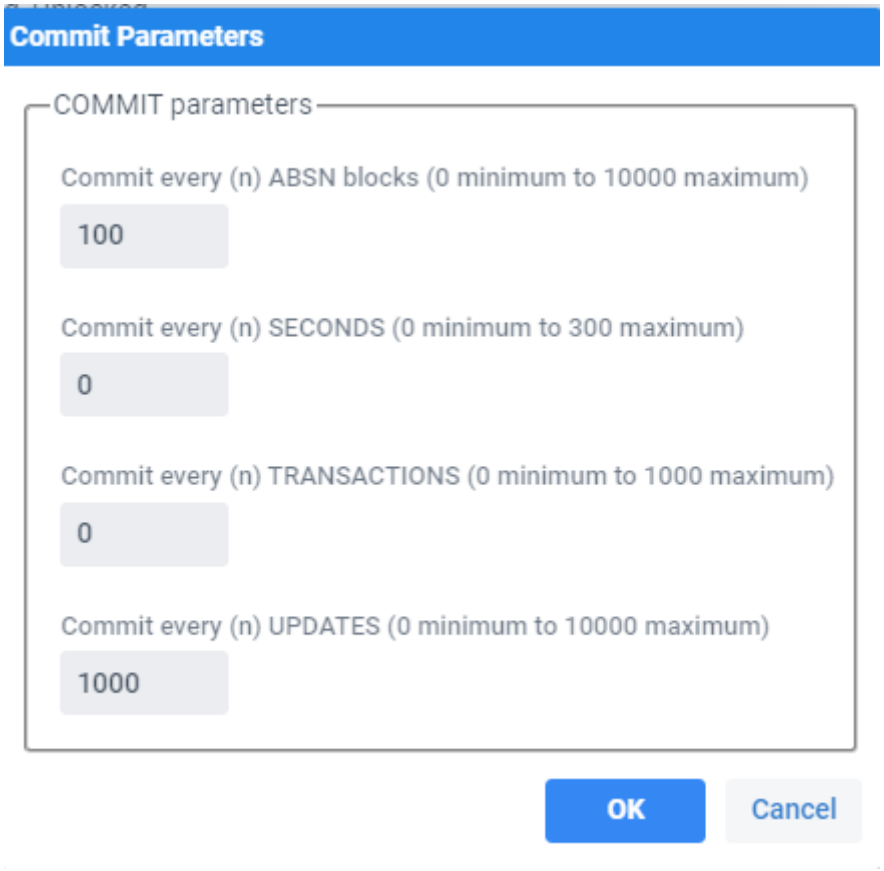
The **Abort** command allows the operator to stop the Client by forcing it to reset the connection to the server.

When this command is selected, a window that requests confirmation of the **Abort** command appears. Once the action is confirmed, the run terminates immediately regardless of whether it is processing updates or doing data extractions. Any uncommitted updates will be rolled back. If the

Client was in the data extraction phase, all data sets that were currently being cloned will have a `ds_mode` of 0, which will cause them to be re-cloned when the next `process` command is issued.

8.4.3 Commit Parameters

The **Commit Parameters** command allows the operator to dynamically change the commit frequency parameters being used by the Client. The changes only remain in effect for the duration of the run. If the `batch_job_period` parameter is enabled in the Client configuration file, changes made to the commit frequency parameters will be overridden when the current period ends. This command opens the following dialog, which is initialized using the Client's effective commit parameters.



Use the supplied input fields to change any of the parameters, and select **OK** to confirm the changes. Confirmed changes will be updated in the next commit. If you do not wish to modify the commit parameters, push the **Cancel** button to close the dialog without making any changes.

8.4.4 Switch Client Log

Selecting **Switch Client Log** causes the Client (**DBClient**) to close the current log file and open a new one.

The file name(s) include the date when the file is created (for example `db20210722.log`). If the old log file was created on the same day as the new file, the new file name will also contain the current

time to make the name identifiable (for example `db20210722_144825.log`). When switching to a new log file the last line of the old log file will contain a line with the text:

```
14:48:25 Log file switched to "logs\\db20210722_144825.log" (Operator Keyin)
```

8.4.5 Switch Trace

Selecting **Switch Client Trace** causes the Client (**DBClient**) to close the current trace file and open a new one.

The file names include the date when the file is created (for example `trace20210722.log`). If the old trace file was created on the same day as the new file, the new file name will also contain the current time to make the name identifiable (for example `trace20210722_144825.log`). When switching to a new trace file the last line of the old trace file will contain a line with the text:

```
Trace file switched to "trace20210726_091904.log" (Operator Keyin)
```

8.4.6 Client Statistics

The **Statistics** command allows the operator to see the current performance statistics for the Client run. The output is displayed in a new page that has 3 tabs (**Run**, **Client**, and **Statement**).

The **Run** tab contains the performance statistics, which are the equivalent of the `"pstats"` command in the command-line client. The **Client** contains the client status, which is equivalent to the `"status"` commands in the command-line client. Finally, the **Statement** tab contains the statement statistics, which are equivalent to the `"astats"` command in the command-line client.

8.5 Data Source Actions

From the menu bar on the data sources page the following commands can be run from the **Actions** drop-down button 

.

8.5.1 Process

Use the **Process** command to clone a data source and track updates. Before cloning a data source, make sure that you've completed all customizations to the data source.

To run a Process command

- Navigate to the data source view of the Client Manager and select the data source for which you want to run the command.
- From the **Actions** button on the menu bar, select **Process**. The command output will be displayed in the data source's console pane.

During the initial clone, DMSII data for the data source is extracted and cloned to the relational database.

After the data source is cloned, you can run the **Process** command as needed to update the relational database. The **Process** command tracks DMSII updates by reading the audit trail and synchronizes the received data to mirror the DMSII database. To schedule updates to occur automatically, use the **Schedule** parameters by navigating to **Settings** > **Configure** > **PROCESSING** > **Schedule**. For more information, see Schedule options.

8.5.2 Generate Scripts

The **Generate Scripts** command creates a set of script files and puts them in the dbscripts subdirectory of the data source's working directory. These script files include SQL scripts that the Client uses to create tables, indexes, drop tables, and remove false duplicate records from tables. Also included, are command files (shell scripts for UNIX clients) that run bulk loader tasks. (Approximately six script files are created for each data table.)

To generate scripts for a data source

- Navigate to the data source view of the Client Manager and select the data source for which you want to run the command.
- From the **Actions** drop-down, select **Generate Scripts**. The command's output is displayed in the data source's console pane.

8.5.3 Reorganize

This command is run following a **Define/Redefine** command for an existing data source when a layout change is detected for the tables in the relational database. This usually happens after a DMSII reorganization that changes the layout of one or more data sets.

The command creates a series of scripts that should be examined before being run, as they may take a long time to run when large tables are involved. The **Reorganize** command implements the second phase of processing the reorganization, which runs the script that alters the table so the layouts match those that would be created using the new DMSII layouts. This is started by running a **Generate Scripts** command to update the scripts. Then, it goes through all the data sets that have been affected by layout changes and runs the reorg scripts created by the **Redefine** command for all the tables that are derived from these data sets. Once this is completed, stored procedures are refreshed, after which the data source should be ready to resume change tracking.

To run the reorganize command for a data source

- Navigate to the data source view of the Client Manager and select the data source for which you want to run the command.
- From the **Actions** drop-down, select **Reorganize**. The command's output is displayed in the data source's console pane.

8.5.4 Define/Redefine

This is a smart command, which combines the command-line Client's `define` and `redefine` commands into a single command. Using the **Customize** command is normally not compatible with using this command. If you use this command to create a new data source, you can use the **Customize** command, as long as user scripts are not involved in the `define` command and the data source was not cloned.

Define/Redefine and Generate Scripts Commands

Once you have created a data source using the **Customize** command, this command will operate just like the **Customize** command in terms of preserving customizations.

Define/Redefine

The **Define/Redefine** command reads the DMSII information from the data source and starts building the relational database tables, layouts, and control information for the data source. If the **Define/Redefine** command is run after the relational database is created, it determines what has changed on the DMSII side and attempts to match it in the relational database while preserving the existing information (this is equivalent to a command-line Client `redefine` command).

To define a data source

- Navigate to the data source view of the Client Manager and select the data source for which you want to run the command.
- From the **Actions** drop-down, select **Define/Redefine**. The command's output is displayed in the data source's console pane.

If the data source was not previously defined you will see that **Not defined** will disappear in the status column for the data source in the data source view.

8.6 Advanced Options and Commands

Getting there

1. From the **Dashboard** select the **Client Manager** server.

   -or-

   Select **Servers** from the left side navigation and select the Client Manager server.

2. From the **Client Manager** data source page, select the **Advanced** drop-down button and choose from the commands and options listed below on this page.

8.6.1 Trace and Log Options

To open the **Trace and Log Options** settings page, select the **Advanced** button drop-down

and select **Trace and Log Options**. Use the options in this dialog box to enable tracing.

Trace files are primarily used by Micro Focus Customer Support for diagnostic purposes. Trace files are not enabled by default because they use a significant amount of hard disk space.

It is recommended that you use the default trace options when sending a trace to Micro Focus Customer Support, unless you are specifically asked to use different options. You can also specify default trace options for a `process` or `clone` command by using **Data Source page > Advanced > Process (with options)** (just check the **Enable default tracing** option in the dialog box).

Note

When you enable tracing options using this dialog, they are applied to the next command or run launched from the Administrative Console unless there is an active **Process** or **Clone** command being run. When there is an active run, the specified tracing options get enabled for the run by sending it an RPC that enables the tracing. Once the tracing options get applied to a run or command, these settings are reset. This eliminates the possibility of having all subsequent commands and runs launched by the Administrative Console execute with tracing enabled.

For more information about trace files, see **Trace and Log Files** in the *Databridge Client Administrator's Guide*.

Global Selection Options

Select all	Click to select all of the trace options
Select default	Click to select the default trace options
Select none	Click to deselect all of the trace options

Trace Options	
Log output	Writes log messages (in addition to trace information) to the trace.log file. (Decimal value: 1 Hexadecimal value: 0x1.)
Server messages	Enables tracing of Databridge Server communications messages for key actions associated with Databridge on the host, including RPCs such as DB_SELECT and DB_READ and their responses. (Decimal value: 4. Hexadecimal value: 0x4.)
Database API calls	Enables the logging of calls from the Databridge Client to the ODBC or OCI APIs. This is also referred to as relational database API tracing. (Decimal value: 16. Hexadecimal value: 0x10)
TCP/IP data	Enables a protocol trace of the information exchange between Databridge Server on the host or Enterprise Server and the Databridge Client. The blocks of data are traced as they are read and written to the TCP interface. The messages are listed in DEBUG format, which is an offset followed by 16 bytes in hexadecimal, followed by the same 16 bytes interpreted as EBCDIC text. The non-printable EBCDIC characters are displayed as periods (.). (Decimal value: 64. Hexadecimal value: 0x40.)
Debug	Enables debugging code that may be added to engineering releases of the Databridge Client. (Decimal value: 256. Hexadecimal value: 0x100.)
Remote console RPC's	Enables remote console message tracing when running DBClient or DBClntCfgServer in conjunction with the service. (Decimal value: 1024. Hexadecimal value: 0x400.)
Callback exit	Enables a trace that helps determine if delays occurred while SQL statements were being executed or while the Databridge Client was waiting for data from the Databridge Server. (Decimal value: 4096. Hexadecimal value: 0x1000.)
Verbose trace	Enables the tracing of certain entries that are normally not traced as the information they provide is often of rather dubious value.

Trace Options	
	(Decimal Value: 65,536. Hexadecimal value: 0x10000)
DMS buffers	Enables the tracing of the allocation and freeing of DMSII buffers. Use this trace when trying to debug a DMSII buffer leak. (Decimal Value: 262,144. Hexadecimal value: 0x40000)
Buffer sizes	Enables a trace that Shows the size calculations for the SQL and host variable buffers. This trace is only useful when you have problem with buffer or host variable overruns. (Decimal Value: 1,048,576. Hexadecimal value: 0x100000)
SQL commands	Enables a trace of SQL commands. (Decimal value: 2. Hexadecimal value: 0x2.)
Control table load	This is load tracing. Load tracing logs information on the Databridge Client control tables as they are loaded from the relational database. (Decimal value: 8. Hexadecimal value: 0x8.)
Bulk loader data	This is a trace of the records that are written to temporary data files (or UNIX pipes) and used by the bulk loader utility during the data extraction phase of cloning. (Decimal value: 32. Hexadecimal value: 0x20.)
Remote console data	This is a protocol trace of the information exchange between DBClient or DBClntCfgServer and the service. The output looks like a Databridge Server protocol trace, except for the fact that all the data is ASCII. (Decimal value: 128. Hexadecimal value: 0x80.)
List config file	This option displays the configuration file parameters as they are processed. (Decimal value: 512. Hexadecimal value: 0x200.)
User script SQL	Temporarily enables SQL tracing when running user scripts during define and redefine commands.

Trace Options	
	(Decimal value: 2048. Hexadecimal value: 0x800.)
DOC Records	Enables a trace of DOC records, which must be enabled. DOC records help get a better insight of what is going on in the audit files. This option is meaningless when Server Message tracing is enabled, because that trace includes DOC records. (Decimal Value: 8192. Hexadecimal value: 0x2000)
Thread Trace	Enables a trace that shows when thread are started and stopped, this trace is seldom useful, unless you are dealing with hung threads. (Decimal Value: 131,072. Hexadecimal value: 0x20000)
Row Count	Enables a tracing of the row counts after SQL statement are executed. (Decimal Value: 524,288. Hexadecimal value: 0x80000)
Trace mask	This entry is automatically filled in as you select trace options. It represents the hex value of the trace mask that is added to the command line using the `/t` option using a prefix of `0x`.
Log Options	
Verbose	Logs some additional information. Most of the verbose output that you had in older versions is now available in the trace options.

8.6.2 Change Database Password

The database password is specified in the authentication settings when you add a new data source. For example, if you use SQL Server authentication, this password is used by the Databridge Client to log in to the relational database.

Some sites periodically require all passwords to be changed, when this is the case, this command allows you to update the database password in the Client configuration file so that the client can continue operating after the database user password was changed.

To change the data source password

1. From the **Client Manager** data sources page, select **Advanced > Change Database Password**.
2. Add the new password in **New password** and the **Verify password** input fields, and then select **OK**.

8.6.3 Change SOURCE Password

Use this procedure to regain access to the DBServer if the password to the SOURCE changes. The SOURCE password is specified by the KEY parameter in the DBServer control file. For more information, see the *Databridge Host Administrator's Guide*.

To change the SOURCE password

1. From the **data sources** page, select **Advanced > Change SOURCE Password**.
2. In the **New Password** box, enter the password exactly as it appears in the DBServer control file
3. In the **Verify Password** box, type the password again, and then select **OK**.

8.6.4 Export Client Configuration

Use this procedure to create a readable copy of the Client configuration file. The Client configuration file contains the current configuration settings for a data source, which are set from the Administrative Console and the Client Manager. (The Administrative Console also includes settings to configure the service. These settings are saved to the service configuration file. For more information about this file, see Client Configuration Files" in **Appendix D** of the *Databridge Client Administrator's Guide*.)

On rare occasions, you'll need to change a parameter that can't be set in the Administrative Console. In this case, you'll need to export the configuration file, edit it, and then import it using the command-line Client **dbutility**.

To export the client configuration file

1. From the **data sources** page, select **Advanced > Export Client Configuration**.

> **Note**
>
> Passwords in the Client configuration file are automatically encoded when using the `export` command to export the file.

A new text data source configuration file (`dbridge.ini`) is created from the binary configuration file (`dbridge.cfg`).

8.6.5 Unload, Reload, and Remove Data Source

This section describes when to use the Unload Data Source, Reload Data Source, and Remove Data Source commands.

Unloading/reloading a data source

The **Unload Data Source** command is used to back up the Client control tables when they're in a good working state. When you use this command, enter a filename, and use optional command-line commands. You can use this back up copy to restore the Client control tables as needed by using the **Reload Data Source** command and entering the same filename used with the **Unload Data Source** command.

The file created by the **Unload Data Source** command is intended only for the program's use. When you report a problem to Micro Focus Customer Support, they may ask you to provide an unload file.

Removing a data source

The **Remove Data Source** command deletes the data tables (that is, the cloned data) and the Client control table entries of a data source. After you select the command, you have the option to remove the working directory and all associated files for the data source. It is recommended that the data tables are deleted after testing configuration features on a new data source before cloning for a production environment.

It is recommended to delete just the data tables after testing configuration features on a new data source or before cloning it for real-world production.

8.6.6 Process (with options)

The **Process With Options** command opens a dialog that allows the user to add command-line options to the **DBClient** `process` command before sending the request to launch a `process` command. the **Process** command in the **Actions** drop-down for data sources launches the process command with no command-line options.

You can select as many options as needed by moving the enabling the corresponding options. When you are done, select **OK** to launch the **Process** command.

Getting there

- From the **Client Manager** select **Advanced** > **Process (with options)**

Options

TOGGLE "DEFER FIXUPS" PARAMETER (-c OPTION)

Adds the `"-c"` option to the **DBClient** command-line, which causes the Client to toggle the `defer_fixup_phase` parameter. When the resulting value of `defer_fixup_phase` parameter is True (enabled), the `process` command does not enter the fixup phase at the end of data

extraction. Instead of issuing a request to the Databridge Server to initiate the fixup phase, the Databridge Client terminates. The next `process` command then picks up where the clone command left off.

USE BCP FOR DATA EXTRACTIONS (-L OPTION)

(SQL Server only) Adds `"-l"` option to the **DBClient** command-line, which forces the Client to use the **bcp** utility instead of the BCP API to load the tables during the data extraction phase.

TOGGLE "USE DBWAIT PARAMETER (-W OPTION)

Adds the `"-w"` option to the **DBClient** command-line, which causes the Client to toggle the `use_dbwait` parameter. When the resulting value of `use_dbawait` parameter is True, the Client uses the DBWAIT RPC and the DBREAD RPC when it is set to False. The DBREAD RPC returns an Audit Unavailable status to the Client, which stops the replication when the Databridge Engine reaches the end of the current audit file or if it cannot open the current audit file as the Engine READ ACTIVE AUDIT parameter is set to False. The DBWAIT RPC enters a wait and retry loop instead.

RECLONE ALL DATA SETS WITH MODE 11 OR 12 (-Y OPTION)

Adds the `"-y"` option to the **DBClient** command-line, which causes the Client to re-clone data sets whose `ds_mode` column in the DATASETS Client control table is set to 10, 11, or 12.

RECLONE ALL ACTIVE DATASETS (-Y OPTION)

Adds the `"-Y reclone_all"` option to the **DBClient** command-line, which causes the Client to re-clone all data sets whose `active` column is 1 in the DATASETS Client control table.

ENABLE DEFAULT TRACING (-D OPTION)

Adds the `"-d"` option to the **DBClient** command-line, which enables default tracing for the `process` command.

> **Note**
>
> If Micro Focus Customer Support asks you for a trace, use this option, unless you are told otherwise.

RUN COMMAND IN READ-ONLY MODE (-Z OPTION)

Adds the `"-z"` option to the **DBClient** command-line, which runs the process command in read-only mode. No updates are made to the relational database. This command is useful if you want to determine how fast the client can run. If you are not sure why the replication is running slow, this option provides a way to determine if the overhead is in the relational database or somewhere else.

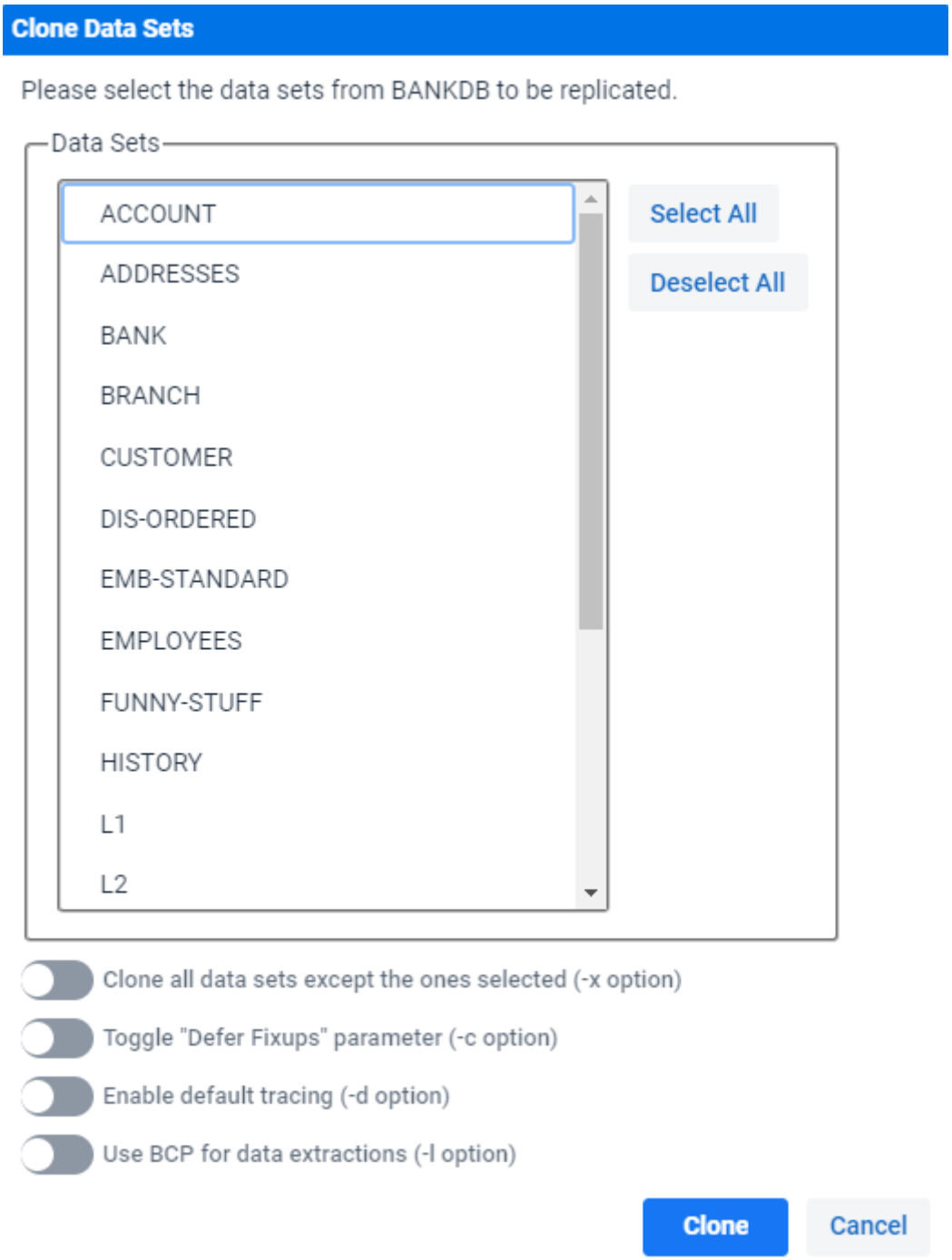
> **Warning**
>
> This option is intended for troubleshooting in a test environment. Do not enable this option in a production environment.

8.6.7 Clone Data Set

Use the **Clone Data Set** option to clone specific data sets in a data source and track updates. Before you clone a data source, make sure that you've completed all customizations to the data source.

Steps to Clone Data Sets

1. In the **data sources** page of the Client Manager, select **Advanced** > **Clone Data Sets** from the drop-down menu.
2. Select the available data sets that will be cloned from the list and set additional parameters as needed.


Clone Data Sets
Please select the data sets from BANKDB to be replicated.
Data Sets
ACCOUNT
ADDRESSES
BANK
BRANCH
CUSTOMER
DIS-ORDERED
EMB-STANDARD
EMPLOYEES
FUNNY-STUFF
HISTORY
L1
L2
Select All
Deselect All
Clone all data sets except the ones selected (-x option)
Toggle "Defer Fixups" parameter (-c option)
Enable default tracing (-d option)
Use BCP for data extractions (-l option)
Clone
Cancel

Additional Settings	Description
Clone all data sets except the ones selected (`-x` option)	When enabled, the Databridge Client clones all active data sets that are not selected in the data set list view.
Toggle "Defer Fixups" parameter (`-c` option)	Toggles the `defer_fixup_phase` configuration file parameter. When you use this option, the dbutility `clone` does not enter the fixup phase at the end of data extraction. Instead of issuing a request to the Databridge Server to initiate the fixup phase, the Databridge Client terminates. The ds_mode values of all cloned data sets remain set to 1 with all of the necessary stateinfo stored in the Client control tables (for example, audit_filenum, audit_block, and host_info). The next `process` command then picks up where the `clone` command left off.
Enable default tracing (`-d` option)	This option enables full tracing.
Use BCP for data extractions (`-l` option)	(SQL Server only) Enable this parameter to force the client to use the bcp utility instead of the BCP API.

3. Select **Clone**.

   During the initial clone, DMSII data for the data source is extracted and cloned to the relational database.

   After the data source is cloned, you can run the **Clone Data Set** command anytime you want to update the relational database. The **Clone Data Set** command tracks DMSII updates by reading the audit trail and synchronizes the received data to mirror the DMSII database. To schedule updates to occur automatically, use the Scheduling parameters in the **Configure** command. For more information on scheduling see the Processing - Schedule.

8.6.8 Log Control Tables

This command is the equivalent of the command-line Client's `display` command. It writes the control tables entries to the **DBClntCfgServer** program's log file for the data source. The log file resides in the data sources **logs** folder and has a name of "*prefix*_cfg*MMDDYYY*.log". Where *prefix* is "db" unless you it has been changed using the **Configure** command in the **Settings** menu for the data source, and *MMDDYYYY* is the date when the log file was created. See the *Databridge Client*

Administrator's Guide for a description of the Client control tables and a list of the abbreviations used in the `display` command output.

None of the output is directed to the Administrative Console as the display command reports are large.

8.6.9 Redefine (with options)

The **Redefine (with options)** command opens a dialog that allows the user to add command-line options to the **Redefine** command before sending the request to run a `redefine` command to the **DBClntCfgServer** program. The **Define/Redefine** command in the **Actions** drop-down on the data source pages launches the **Redefine** command with no command-line options.

Getting there

- From the **Client Manager** select **Advanced** > **Redefine (with options)**

Options

REDEFINE ALL DATA SETS (-R OPTION)

Forces all data sets to be redefined. Use this option if you made a change using the **Customize** command that requires all data sets to be redefined.

TOGGLE USE OF USER SCRIPTS (-R OPTION)

Enables the parameter `use_dbconfig`. If this parameter is enabled, the command will revert to using user scripts. Use this option if you have created the users scripts by running the **Create User Scripts** command from the **Advanced** button drop-down for the data source and customizations are not working as expected. Running a **Redefine** command with this option and the **Redefine all data sets** options would restore the control tables to their original state from the **Create Scripts** command.

OVERRIDE WARNINGS (-U OPTION)

Enabling this option will override conditions that the **DBClient** would otherwise interpret as a possible user error. This can happen when mixing **Customize** and **Redefine** commands. It is safe to run a **Redefine** command if you have not run a **Customize** command since the last time a **Process** command was run to process updates. If you get into a situation where the **Redefine** cannot be ran, as the unload file used to back up the Client Control table was created by the **Customize** command, using this option will solve the problem. However, any customization made since the unload file was created will be lost.

The reason for this difference is that the **Customize** command can be run on demand, but the **Redefine** command does not allow you to run the command more than once, unless you restore the control tables from the unload file it creates. Starting with a **Customize** command causes the **Redefine** command to restore from a backup, which would lead to lost updates from the **Customize** command.

8.6.10 Generate All Scripts

Enabling this option initiates a **Generate Scripts** command `"-u"` option on the command-line, which causes the command to generate the scripts for all the tables to be generated. This command has no dialog, as you can add a single option to the command. The **Generate Scripts** command in the **Actions** button drop-down for the data source only generates the scripts for the tables whose scripts are not current.

Getting there

- From the **Client Manager** select **Advanced** > **Generate all scripts**

8.6.11 Refresh Data Sets

Getting there

- From the **Client Manager** select **Advanced** > **Refresh Data Set**

Enabling this command should not be common. When enabled, this command will drop the stored procedures associated with the tables mapped from a data set. The **Reorganize** command is closely related to this command, as it is used when the relational database tables' layout has changed as a result of a DMSII reorganization, or from user initiated changes that affect the table layout. The most common situation where you would need this command is when a user alters the stored procedures associated with the tables for a specific data set.

This command can be applied to a single data set or all data sets. When refreshing more than one data set you will need to run the command multiple times.

Options

REFRESH ALL DATA SETS

Select this option to refresh all data sets.

REFRESH DATA SET

Select a specific data set to refresh by using the drop-down list of data sets.

8.6.12 Reorg (Multi-source, 2nd source)

This is a special form of the **Reorganize** command used for multi-source data sources with merged tables. This command allows multiple DMSII databases whose DASDLs are identical to be replicated to a single relational database. When DMSII reorganizations occur in these environments, you cannot run a **Reorganize** command that alters the tables for each data source, as there is only one set of tables. Ensure that all the Clients for the various DMSII data sources are caught up with the reorganizations before you can run a **Reorganize** command. Once the command is run the tables will be altered. This command adds the `-n` option to the command-line

for the command, which inhibits the running of scripts to alter the tables, but does everything else the **Reorg** command does. Refer to the *Databridge Client Administrator's Guide* for more information.

8.6.13 Create and Run User Scripts

You can save customizations to the relational database table layout as user scripts. If the system fails, the `define` command automatically uses these user scripts to rebuild the client control tables.

When modifying relational database table layouts from the **Customize** option, it is recommended that changes are saved as user scripts for backup purposes. If a hard disk or system failure occurs, user scripts can be used to recover customized table layouts.

After the console copies the existing user scripts to the user scripts archive directory, it creates a new set of user script files in the user scripts directory. This information can be seen in the console output located at the bottom of the Administrative Console.

> **Note**
>
> You do not need to archive user scripts from previous versions of the Databridge Client/ Administrative Console. The Windows installer provides you with an option of copying the configuration files and user scripts from the previous version's working directory to the new working directory.

To create user scripts from the Administrative Console

- With the data source selected in the **data sources** page, select **Advanced** > **Create User Scripts**.

  Following a customization of a data set, configure the user script with the options below and select **OK**.

  The script will be saved in the **scripts** directory.

DO NOT BACKUP OLD USER SCRIPTS

When enabled, this command removes all the customization user scripts and recreates them while preserving any other scripts that the user might have created (for example cerate table scripts).

INCLUDE DATA SOURCE NAME IN USER SCRIPTS

Enable this option to include the selected data source name in the user scripts.

Run User Scripts

To run a user script

- With the data source selected in the **data sources** page, select **Advanced** > **Run Script**.

8.6.14 Run History

When enabled, the monitor will maintain a table with the run history of the last 10 runs. This includes the start and end times for the runs, the ending state information values, and the exit codes. This command is only available when there is no active run.

To see the Run History

- With the data source selected in the **data sources** page, select **Advanced** > **Run History**.

9. Customizing Data Sets

9.1 Customizing Data Sets

Use the **Customize** option to map individual DMSII data sets (and their items) and customize the table layout of the relational database.

Customizations should be made before cloning a data source or your changes may require that you re-clone. If you want to make customizations that affect the entire data source, use the **Configure** option available from the **Settings** drop-down button on the data sources page. See Customize the Client Configuration for more information on customizations for the entire data source.

> **Note**
>
> The customize command is a cooperative process between the **DBClntCfgServer** program, the Client Manager service, the code in the Administrative Console server, and the browser. Do not use the back button in the browser or close the browser before you successfully exit the **Customize** command by using the back button and **Done** button provided in the Administrative Console user interface. Closing the browser will leave the **DBClntCfgServer** program in a bad state. When this happens, the program can timeout, or the process can be ended by using the Task Manager.

When **Customize** is selected, the data set view will be displayed. In this view, you can customize data sets, deselect data sets you do not wish to replicate, set data set specific options that override global parameters specified in the Client configuration file, and link virtual and real data sets from which they are derived.

From the data sets view, select a data set to load the **DMS item** tab where customizations on the data sets DMS items can be performed.

DMS Item Customization Examples: - Allow you to disable items that you do not wish to replicate. - Modify how an item is replicated (e.g. a date represented as a NUMBER(8) in DMSII can be replicated as a relational database date) - Items with an OCCURS table can be mapped to a secondary table (or it can be flattened with the table).

Clicking on the **Relational** tab in the DMSII page takes you to the relational view where you customize data tables and data items. The **Done** button in both views takes you back to the data sets view.

Relational Customization Examples: - Rename data tables and data items. - Changing the order of columns in a table. - Modify the data types of an item. - SQL Server Client: - Apply data masking to sensitive items.

To exit the **Customize** command return to the data set view and push the **Done** button in the bottom-right of the Administrative Console.

9.1.1 DMS Items Page

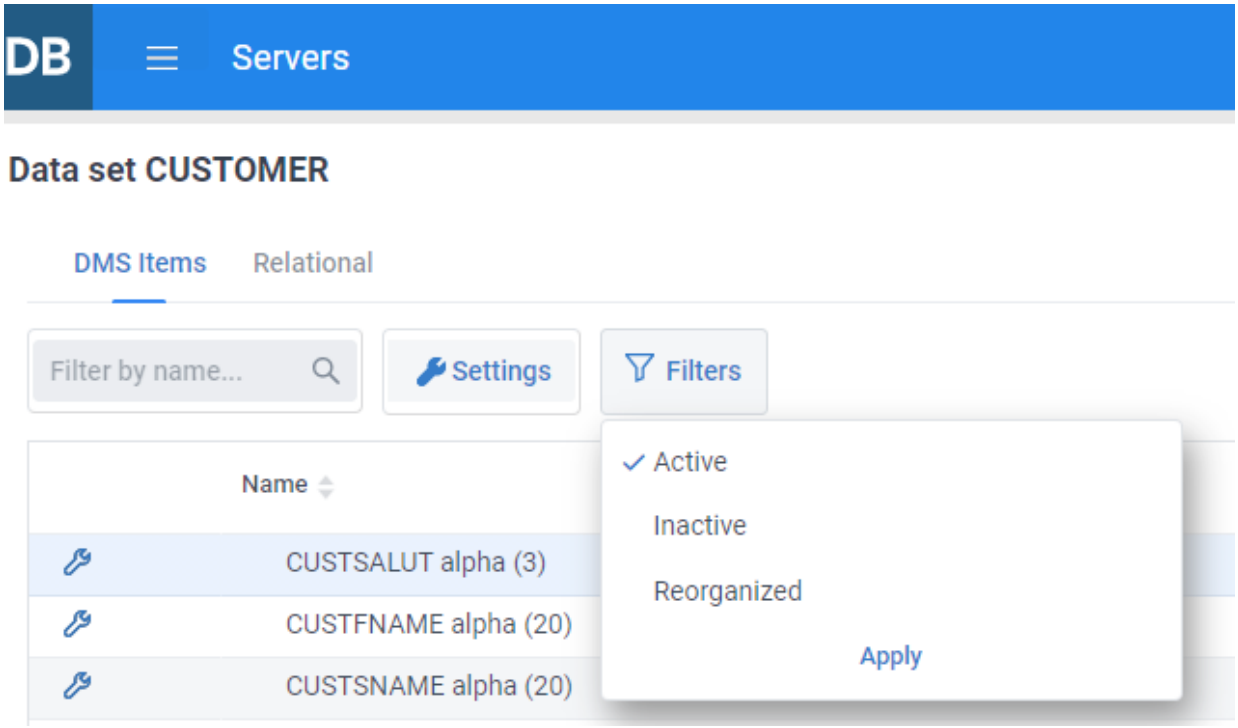
Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (select a data set)

Selecting a data set in the data sets view opens the **DMS items** tab for the data set. The **DMS items** tab is a table of DMS items which has 4 columns. The entries have a **Properties** icon to the left of the DMS item name that allows you to access the properties page for the item. This is followed by the "**Name**" column that contains the item name followed by the data type. The next column named "**Key**", contains a key icon for any item that is a member of the DMSII SET that is being used as the source of the index. The "**Status**" column reflects the value of the `active` column in the DMS_ITEMS Client control table, and the **Customized** column contains a checkmark when the value of a property for the DMS item has been changed.

> **Tip**
>
> Unlike the data sets view, the DMS items view can contain groups that need to be expanded to see the items.

Filters



Since several DMSII data sets typically have hundreds of items, you can filter items to make it easier to locate specific items. Typing a set of characters in the **Filter by name** edit box will only show you the data sets whose names contain the string of characters entered. The **Filters** button, when selected, will present you with 3 filtering choices, **Active** (default), **Inactive** and **Reorganized**.

When the filter is set to **Active**, only DMS items whose `active` column is 1 in the DMS_ITEMS Client control table will be shown. If you want to also see DMS items whose `active` column is 0, select **Inactive** in the **Filters** drop-down. Select **Apply** to view the DMSII items list results from the newly applied filter. The **Reorganized** option is used during the processing of DMSII reorganizations, where the view will be limited to data sets that have been affected by a reorganization. The Administrative Console will automatically enable this filter when it detects that it is processing a DMSII reorganization.

In addition to filtering, you can sort the table by **Name** (which is the default), **Key**, **Status**, or **Customized** by using the up and down arrows in the column headers.

> **Warning**
>
> The view that shows only what has changed from a DMSII reorganization will help prevent unnecessary changes to the layout of a dataset that was not affected by a reorganization.

> **Tip**
>
> For data sets that contain DMSII GROUP items the filtering will descend into the GROUPS, so if you have a naming convention ending the item name with "-DT" when they are dates, using this as a filter will find all the dates.

9.1.2 Customizing DMS Items

How to select a DMS Item

- Do one of the following:

  Select the properties icon  for the desired data item.

  -or-

  Click on the DMS item table row for the desired DMS Item (the table row will be highlighted in blue) indicating that it is selected and select **Properties** from the **Settings** drop-down button in the menu bar.

  Currently, there is not an **Information** page for DMS items, as all of the properties we display are editable.

Once a DMS item has been customized, return to the **Data Sets** page with the DMSII Items tab selected. Select the **Relational** tab to view the tables mapped from the data set in question and their columns. If the Client control tables have not yet been updated with the changes you made, pushing this button causes the data sets that need mapping to be mapped. You do not have to do this at this time; when you exit the **Customize** commend by navigating away from it using one of

the provided links, this operation will be automatically performed. The Relational tab has a tab named **DMS Items** that gets you back to this page.

When you are done customizing the data set push the **Done** button to return to the data sets view. If you navigated to the relational view, you will be able to return to the DMSII item tab view or to the data sets view.

9.1.3 Relational Page

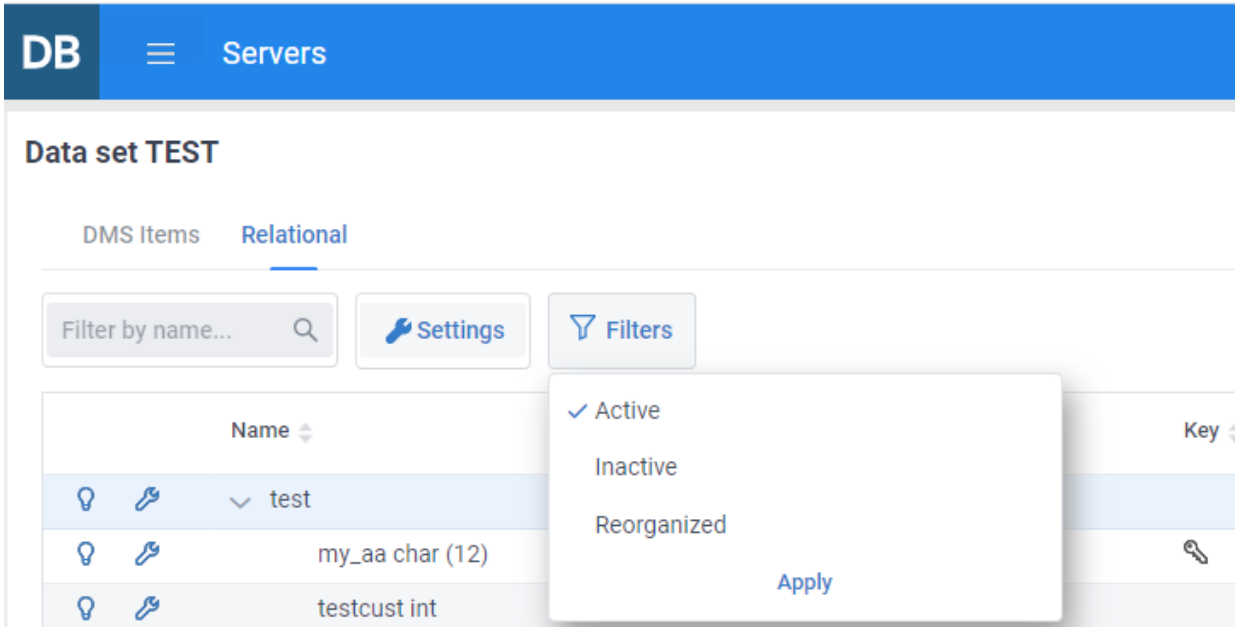
Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (click on data set) > **Relational**

Selecting the **Relational** tab in the DMS items view opens the Relational tab for the data set. The relational tab is a table of objects, which has 4 columns. The term "object" will be used to denote data tables and data items throughout this section. The entries have two icons to the left of the object name that allow you to access the **Properties** and **Information** pages for the object. This is followed by the **Name** column that contains table name or the column name followed by the data type. The **Key** column and contains a key icon for any item that is part of the index for the table it belongs to. The **Status** column reflects the value of the `active` column in the DATATABLES or DATAITEMS Client control table. Lastly, the **Customized** column contains a checkmark when the value of a property for the object has been changed.

Unlike the DMS items tab, the Relational tab is grouped by table, as there can be more than just one table associated with a data set.

Filters



Since several DMSII data sets typically have hundreds of items, you can filter items to make it easier to locate specific items. Typing a set of characters in the **Filter by name** edit box will only

show you the data sets whose names contain the string of characters entered. The **Filters** button, when selected, will present you with 3 filtering choices, **Active** (default), **Inactive** and **Reorganized**. When the filter is set to **Active**, only DMS items whose `active` column is 1 in the DMS_ITEMS Client control table will be shown. If you want to also see DMS items whose `active` column is 0, select **Inactive** in the **Filters** drop-down. Select **Apply** to view the DMSII items list results from the newly applied filter. The **Reorganized** option is used during the processing of DMSII reorganizations, where the view will be limited to data sets that have been affected by a reorganization. The Administrative Console will automatically enable this filter when it detects that it is processing a DMSII reorganization.

In addition to filtering, you can sort the table by **Name** (which is the default), **Key**, **Status**, or **Customized** by using the up and down arrows in the column headers.

> **Warning**
>
> The view that shows only what has changed from a DMSII reorganization will help prevent unnecessary changes to the layout of a dataset that was not affected by a reorganization.

> **Tip**
>
> For data sets that contain DMSII GROUP items the filtering will descend into the GROUPS, so if you have a naming convention ending the item name with "-DT" when they are dates, using this as a filter will find all the dates.

9.1.4 Customizing Data Tables and Data Items

How to select a Data Table or Data Item

- Do one of the following:

  Select the properties icon  for the desired data table or data item.

  -or-

  Click on the desired data item or data tables row (the table row will be highlighted in blue) indicating that it is selected. Select **Properties** from the **Settings** drop-down button on the menu bar.

Once you are done customizing an object, you will return to this page when you close the properties page by pushing one of the two buttons in the bottom-right corner of the page. Once you are done customizing the items in the relational view, you can select the **DMS Items** tab to return to the DMS items view, or you can select the **Done** button to return to the data sets view.

9.2 Data Set Properties

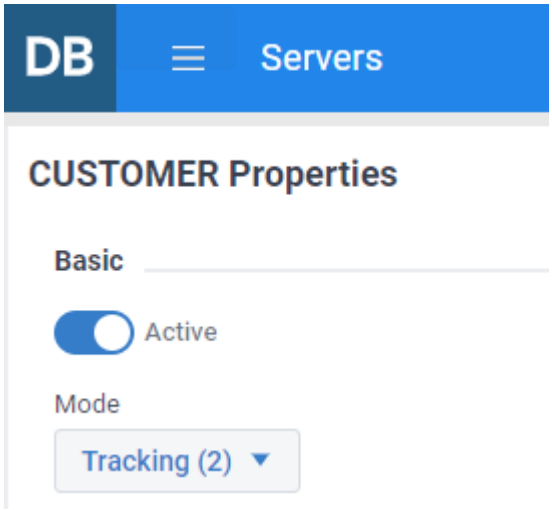
Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (select data set) > **Settings** > **Properties**

The customizable properties for a data set are organized in the following sections:

- Basic
- DMSII Information
- Options
- Virtual dataset
- User Columns

9.2.1 Basic Properties



This section has two properties: **Active** and **Mode**, which correspond to the `active` and `ds_mode` columns in the DATASETS Client control table.

Disabling **Active** (toggle will turn gray) has the effect of disabling the mapping of the data set to the relational database. This is different than disabling **Active** in the **Settings > Data Sets** page for the data source. When disabled, the specific data set will stop the replication for the data set in question. Enabling the **Active** option will cause the data set to be mapped.

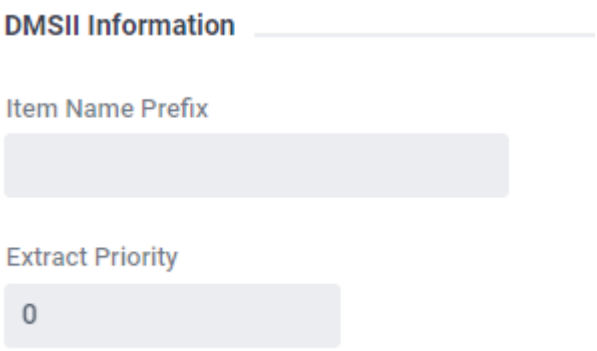
> **Note**
>
> Disabling **Active** for a data set that has customizations on the relational database will result in changes being lost.

Changing the **Mode** option to a different value is not common while customizing except in some special situations. See below for circumstances that would permit changing this option.

- Setting the **Mode** to Clone (0) to force the data set to be re-cloned. Do this if reorganizing will take longer than re-cloning.
- Following a failed `Reorganize` command you may want to correct the reorg script that failed. Change the **Mode** from `Reorg Failed (33)` to `Reorg Needed (31)` and rerun the `Reorganize` command, which can be restarted.

You would normally run these commands from the **Settings > Data Sets** menu for the data source. We provide them in the **Customize** command in case you need to do this while customizing.

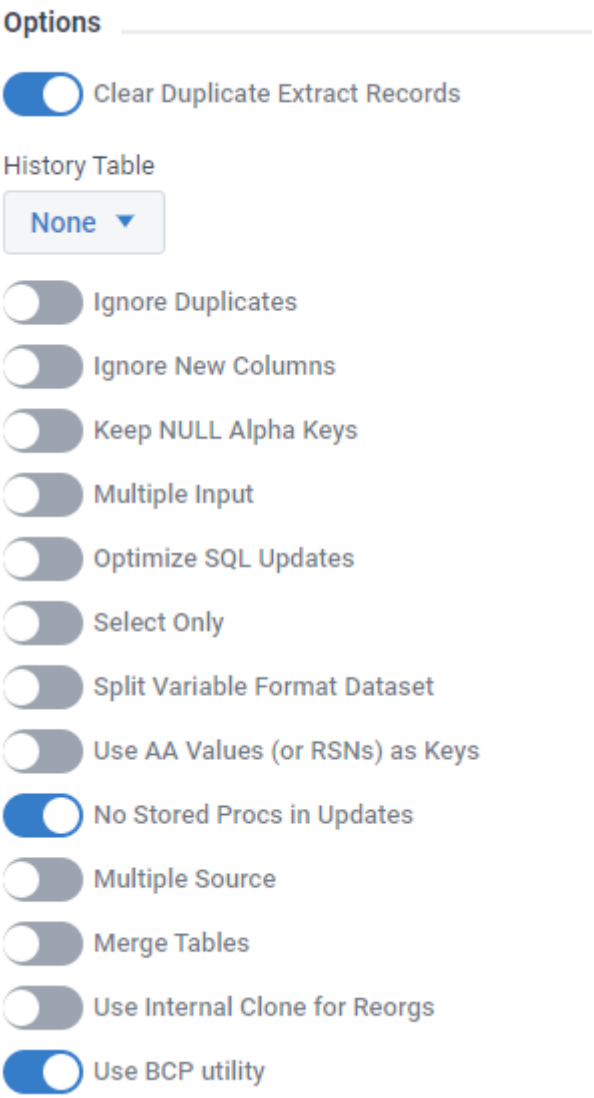
9.2.2 DMSII Information



This section contains the two configuration options **Item Name Prefix** and **Extract Priority**. These options correspond to the values of the `item_name_prefix` and `extract_priority` columns in the DATASESTS Client control table.

- **Item name Prefix** is used to make the Client automatically strip fixed prefixes from data item names when naming the columns in the corresponding tables in the relational database. This avoids having to do massive renames to get rid of redundant prefixes.
- **Extract Priority** allows you to control the order in which items are selected when initializing the connection to the Databridge server. Data sets are extracted in the order in which they are selected. The default, is to select them in ascending structure number and record type order. When `extract_priority` is set to a non-zero value, items with higher `extract_priority` values are selected first and the structure number and record type are used to determine order when duplicates are detected. This allows you to force large data sets (for which the data extraction could take a long time) to be extracted first, as this would create more overlap shortening the duration of the clone.

9.2.3 Options



This section contains a series of options that allow you to change some of the bits in the `ds_options` column of the DATASETS table. The **History Tables** parameter, which controls two mutually exclusive `ds_options` bits is represented by a list box with 3 entries (**None**, **Save History** and **Save History Only** to prevent both bits from being set).

Refer to the DATASETS Client control table section of *Databridge Client Administrator's Guide* for more information about these bits. The various option bits are listed below with their

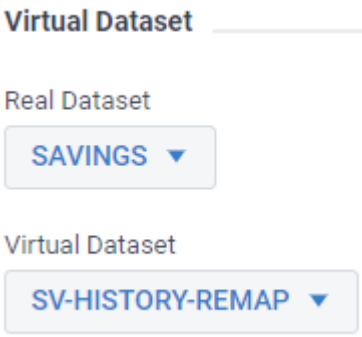
corresponding names used in the *Databridge Client Administrator's Guide* and the configuration file parameter(s) that define their initial values, when applicable.

Name in UI	Name in Client Administrator's Guide	mask	config parameter
Clear Duplicate Extract Records	DSOPT_Clrdup_Recs	32,768	`clr_dup_extr_recs`
Ignore Duplicates	DSOPT_Ignore_Dups	32	`suppress_dup_warnings`
History Tables > Save History	DSOPT_Save_Updates	8	`history_tables = 1`
History Tables > Save History Only	DSOPT_HistoryOnly	8192	`history_tables = 2`
Ignore New Columns	DSOPT_Supp_New_Columns	256	`suppress_new_columns`
Keep NULL Alapha keys	DSOPT_Keep_Null_Alpha_Keys	128	
Multiple Input	DSOPT_MultiInput	512	
Optimize SQL Updates	DSOPT_Use_bi_ai	1	`optimize_updates`
Select Only	DSOPT_Select_Only	64	
Split variable Format Dataset	DSOPT_Use_AA_Only	65,536	`split_varfmt_dataset`
	DSOPT_Use_AA_Only	16,384	

Name in UI	Name in Client Administrator's Guide	mask	config parameter
Use AA Values (or RSNs) as Keys			
No Stored Procs in Updates	DSOPT_No_StoredProcs	4	`use_stored_procs`
Multiple Source	DSOPT_MultiSource	1024	
Merged Tables	DSOPT_MergedTables	2048	
Use Internal Clone	DSOPT_Internal_Clone	4,194,304	`use_internal_clone`
Use BCP utility	DSUPT_Use_BCP	16,777,216	`use_bcp`

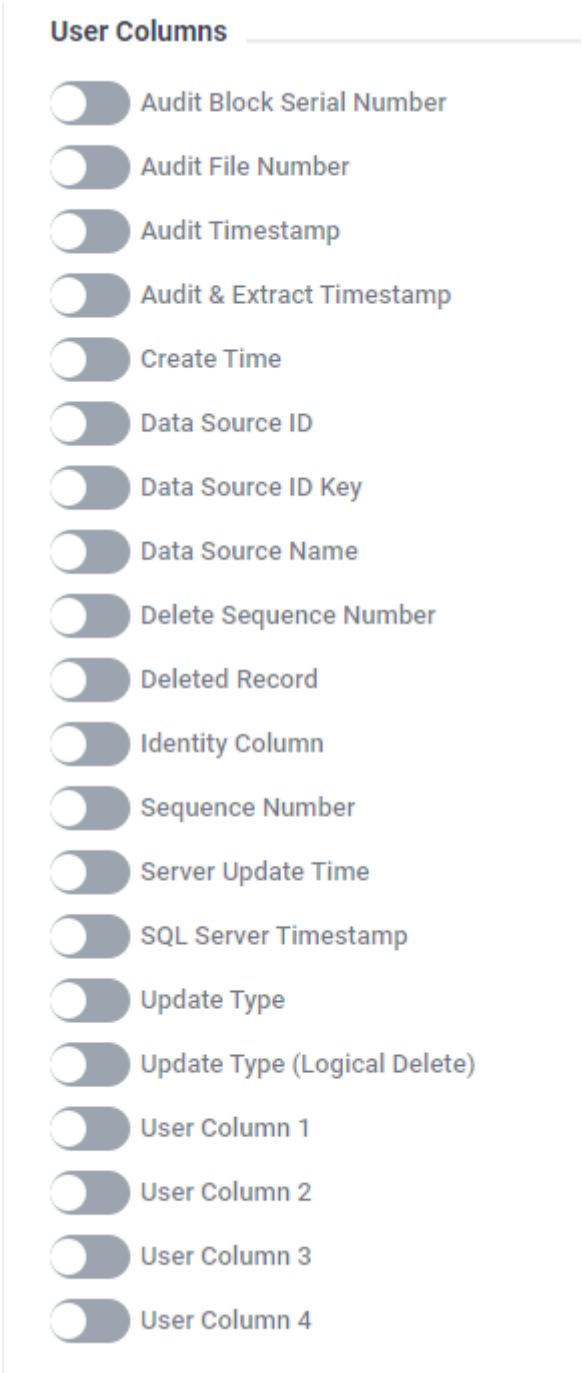
See the DATASETS Client Control table section of the *Databridge Client Administrator's Guide* for details these options. You can change the setting for these bits on a data set by date set basis using customization. You should set the parameter in the configuration file to reflect the most commonly used setting, as this will reduce the amount of changes needed.

9.2.4 Virtual Data Set



This section allows you to link virtual data sets and real data sets from which they are derived using the `virtual_ds_num`, `real_ds_num` and `real_ds-rectype` columns in the DATASETS client control table. The two controls are list-boxes that contain all qualifying data sets that can be selected.

9.2.5 User Columns



This section lists the various user columns that are to be added to this data set. These bit default to the value defined by the `external_columns` parameter in the Client configuration file. This set of options allows you to further customize the user columns to be used for the tables derived from this data set.

9.3 Data Set Information (Read-Only Properties)

Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (select data set) > **Settings** > **Information**

The read-only properties for a data set are organized in the following sections:

- Basic
- DMSII Information
- Miscellaneous Flag
- Options
- Virtual dataset
- User Columns

The editable properties also appear in the Properties page for the data set.

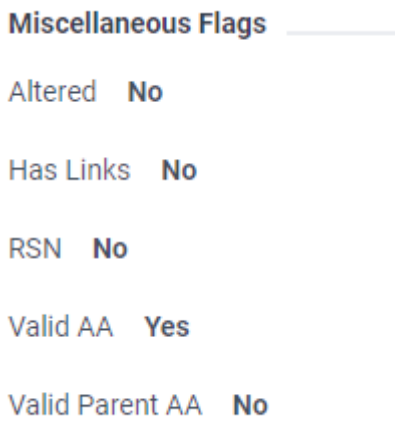
9.3.1 Basic Properties

This section has two properties: **Active** and **Mode**, which corresponds to the `active` and `ds_mode` columns in the DATASETS Client control table. Both properties can be edited from the Data Set Properties Page.

9.3.2 DMSII Information

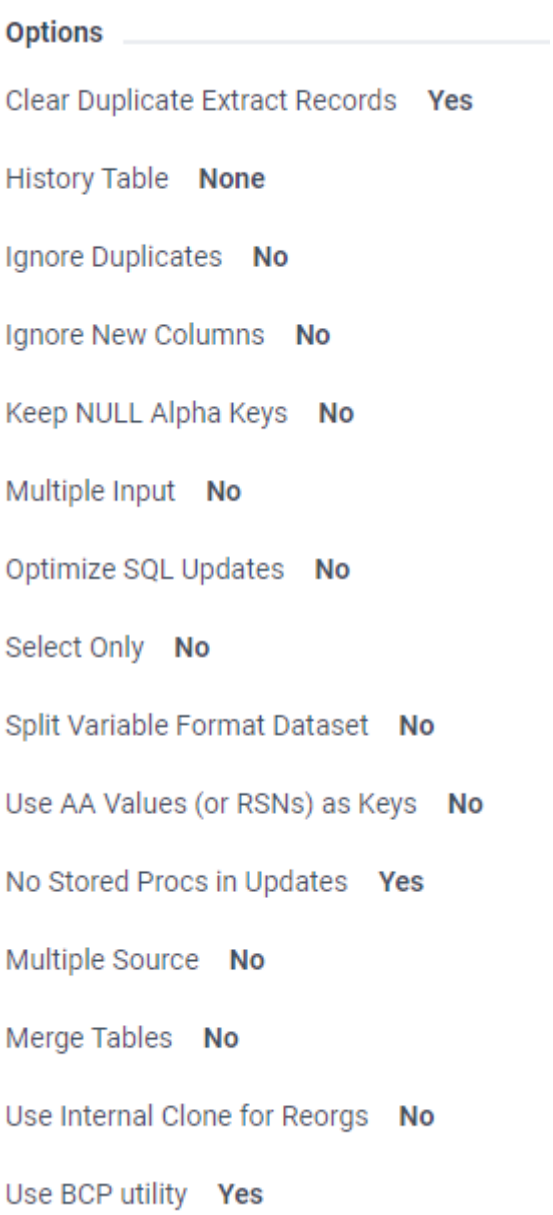
This section contains two items **Item name Prefix** and **Extract Priority**, they correspond to the values of the `item_name_prefix` and `extract_priority` in the DATASESTS Client control table. Both of these can be edited from the Data Set Properties Page.

9.3.3 Miscellaneous Flags



This section contains the values for relevant bits in the `misc_flags` column of the DATASETS Client control table that are useful when customizing the data set. The **Altered** bit (`DSFLG_Altered`) when set, indicates that the data was altered by the Databridge Support library. The **Has Links** bit (`DSFLG_Links`) when set, indicates that the data set has links to other data sets. The **RSN** bit (`DSFLG_Static_AA`) when set, indicates that the Databridge Engine is using RSNs in place of AA Values. The **Valid AA** bit (`DSFLG_Valid_AA`) when set, indicates that the data set has valid AA Values. The **Valid Parent AA** bit (DSFLG_Uses_Parent_AA) when set, indicates that the data set, which is an embedded data set whose parent data set has valid AA Values. The embedded data set uses a column named `parent_aa` to use the parent record AA value as a foreign key.

9.3.4 Options



This section contains a series of options that represents individual bits in the `ds_options` column of the DATASETS table. The **History Tables** parameter, which controls two mutually exclusive `ds_options` is represented by a list box with 3 entries (`None` , `Save History` and `Save History Only` to prevent both bits from being set).

Refer to the DATASETS Client control table in the *Databridge Client Administrator's Guide* for more information about these bits. The various option bits are listed below with their corresponding

names used in the *Databridge Client Administrator's Guide* and the configuration file parameter(s) which define their initial values when applicable.

Name in UI	Name in Client Administrator's Guide	mask	config parameter
Clear Duplicate Extract records	DSOPT_Clrdup_Recs	32,768	clr_dup_extr_recs
History Tables > Save History	DSOPT_Save_Updates	8	`history_tables = 1`
History Tables > Save History Only	DSOPT_HistoryOnly	8192	`history_tables = 2`
Ignore Duplicates	DSOPT_Ignore_Dups	32	`suppress_dup_warnings`
Ignore New Columns	DSOPT_Supp_New_Columns	256	`suppress_new_columns`
Keep NULL Alapha keys	DSOPT_Keep_Null_Alpha_Keys	128	
Multiple Input	DSOPT_MultiInput	512	
Optimize SQL Updates	DSOPT_Use_bi_ai	1	`optimize_updates`
Select Only	DSOPT_Select_Only	64	
Split variable Format Dataset	DSOPT_Use_AA_Only	65,536	`split_varfmt_dataset`
	DSOPT_Use_AA_Only	16,384	

Name in UI	Name in Client Administrator's Guide	mask	config parameter
Use AA Values (or RSNs) as Keys			
No Stored Procs in Updates	DSOPT_No_StoredProcs	4	`use_stored_procs`
Multiple Source	DSOPT_MultiSource	1024	
Merged Tables	DSOPT_MergedTables	2048	
Use Internal Clone	DSOPT_Internal_Clone	4,194,304	`use_internal_clone`
Use BCP utility	DSUPT_Use_BCP	16,777,216	`use_bcp`

See the DATASETS Client Control table in the *Databridge Client Administrator's Guide* for more information. You can change the setting for these bits on a data set on a date set basis using customization. You should set the parameter in the configuration file to reflect the most commonly used setting, as this will reduce the amount of changes needed.

9.3.5 Virtual Data Set

This section allows you to link virtual data sets and real data sets from which they are derived using the `virtual_ds_num`, `real_ds_num` and `real_ds-rectype` columns in the DATASETS client control table. The two controls are list-boxes that contain all the qualifying data sets that are available.

9.3.6 User Columns

This section lists the various user columns that will be added to the data set. These bits default to the value defined by the `external_columns` parameter in the Client configuration file. This set of options allows you to further customize the user columns to be used for the tables derived from this data set.

9.4 DMS Items Properties

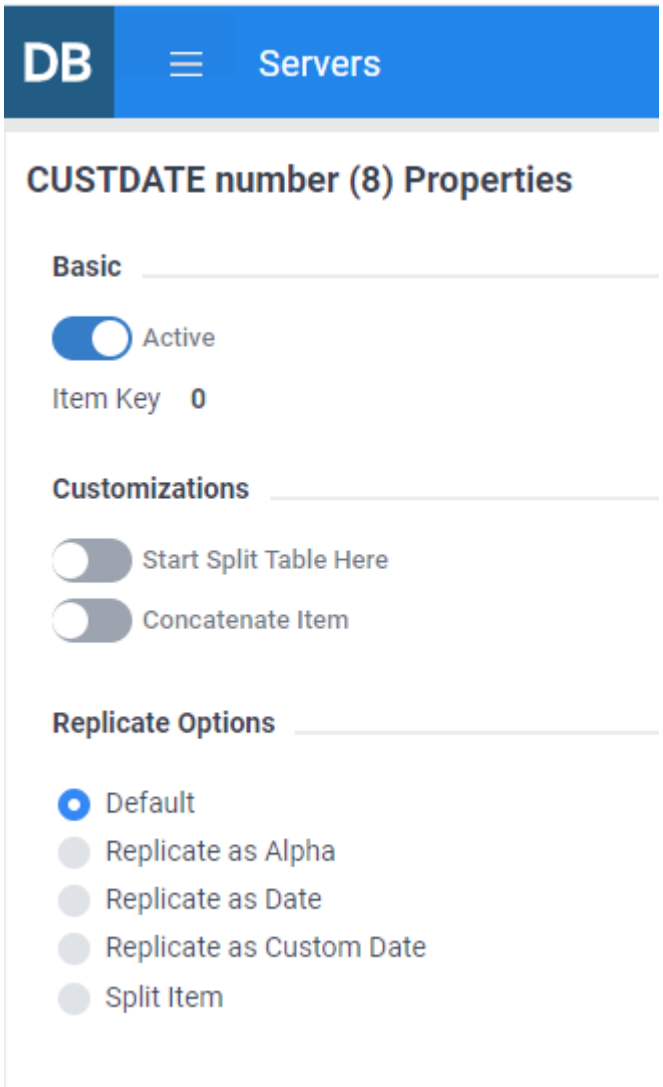
Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (click on data set) > (click on Properties icon)

The customizable properties for a DMS items are organized in the following sections:

- Basic
- Customizations
- Options
- Replicate Options

These properties are dependent on the data type and length of the item and several other factors, so the properties will vary from item to item, as we only show properties that are applicable to the item.



9.4.1 Basic Properties

This section has two properties, **Active** and **Item Key**. **Active** represents the value of the `active` column in the DMS_ITEMS table for the item. This will be True, unless you set it to False (by disabling this option), in which case the item will not be mapped to the relational database tables. **Item_Key** represents the value of the `item_key` column in the DMS_ITEMS Client control table; a non-zero value indicates that the item is a key and it also specifies its position within the SET.

9.4.2 Customizations

This section has properties that deal with data alteration. The supported operations are:

- Merging Neighboring Items
- Concatenating Items
- Redefining GROUPS of Like Items as Single Items
- Initiate a table split after the current item
- End a table split after the current item

Replicate Options

This section contains a set of mutually exclusive replication options. In the above example for a NUMBER(8) item your choices are limited to the 5 shown in the above screen shot.

Default

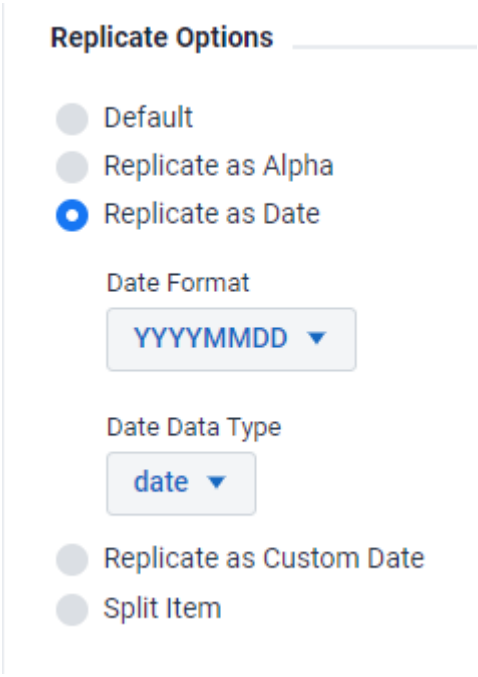
Undoes a previous selection and returns things to their default state.

Replicate as Alpha

The item, which has a DMSII data type of NUMBER)(*n*), will be created as character data in the relational database (i.e. char(8) or varchar(8)).

Replicate as Date

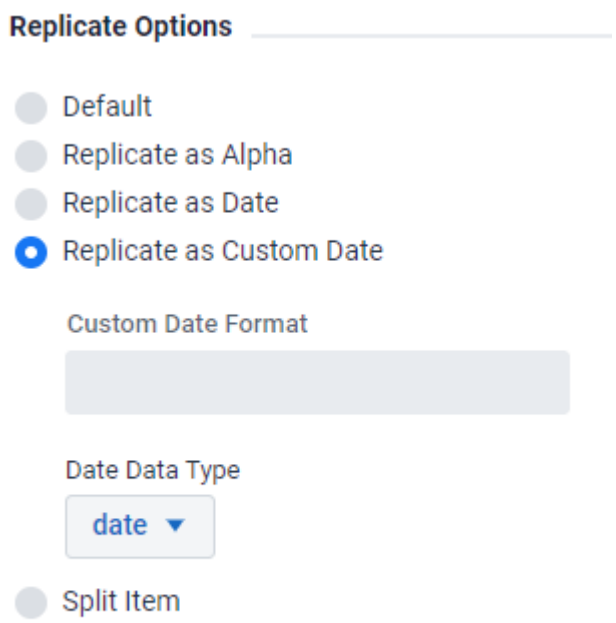
The item will be stored as a relational database date (or timestamp), whose format in DMSII and data type in the relational database may need to be specified, unless the defaults used are correct. The `default_date_format` is used to reduce the need for changing the DMSII date format. When you this option is enabled, input fields will expand as shown is the screen shot below.



Replicate as Custom Date

The item will be stored as a relational database date (or timestamp), whose format in DMSII is non-standard and will need to be entered using a set of code words. The code words for the various entries are:

Entry type	Code	Comment
4-digit year	yyyy	You can only have one year entry
2-digit year	yy	You can only have one year entry
month	mm	
day	dd	
hours	hh	
minutes	mi	
seconds	ss	
decimal point	.	You can add a decimal point before the fraction of second entry
millseconds	fff	You can only have one entry for fraction of seconds
microseconds	ffffff	You can only have one entry for fraction of seconds
centiseconds	ff	You can only have one entry for fractions of seconds



An example of a custom date format for a NUMBER(6) is "yyyymm".

Replicate As Binary

The item will be created as binary data in the relational database (i.e. binary(n) in SQL Server and raw(n) in Oracle). When you enable this option you get an additional option labeled **Translate Binary Data**. If you do not enable this option, the binary data is stored as is. However, if you enable this option the data is translated from EBCDIC to ASCII, in the same manner as normal data is, except all control characters are allowed to be part of the data.

Replicate as 3 Booleans

The item, which is a NUMBER(1) will be created as 3 columns of type boolean (i.e. bit in SQL Server and NUMBER(1) in Oracle). This construct is used in MISER databases to pack flags into a NUMBER(1) item with an OCCURS clause. The items have a range of 0-7, where each bit is represented by a boolean.

Replicate as Number

The item, which is a DMSII date type of ALPHA(n), will be created as a numeric item. This will only make sense if the data is known to only contain numeric characters. If this is not the case, the Client will treat the item as being in error and set its value to NULL.

Replicate as Time

The item will be stored as a relational database time, whose format in DMSII is non-standard and will need to be entered using a set of code words. If the database does not have a time data type it will store as a number using the format "hhmiss".

Replicate as RSNs

The item which has a DMSII data type of REAL, will be treated like an RSN. Typically, this option will be automatically visible for the RSN by the Databridge Engine.

Replicate as GUID

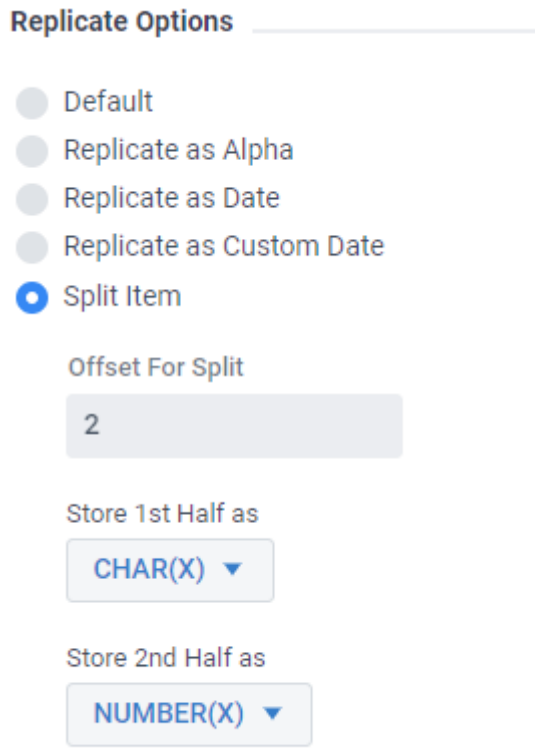
(SQL Server client only) The item which has a DMSII data type of ALPHA(36), and will be stored as a GUID (uniqueidentifier).

Replicate as ResFlag

The item which has a DMSII data type of NUMBER(1), will be treated as a Res Flag. This is only applicable for MISER databases, and is used to distinguish between resident history records and non-resident history records in the various history virtual data sets.

Split Item

The item, which must be either an ALPHA(N) item or an unsigned number (i.e. NUMBER(N)), will be split into two parts when mapped into the relational database table. When this option is enabled, a set of controls are expanded, as seen in the screen shot below. Enable **Offset for Split** to enter where in the items the split is to be made. For example, if you split a NUMBER(14) at offset 2 you get a NUMBER(2) and NUMBER(12). This was used to split off a code contained in a 14-digit account number to a new column that was stored as ALPHA. The last controls allow you to specify how the parts are to be stored. The available options are NUMBER(X) and CHAR(X).



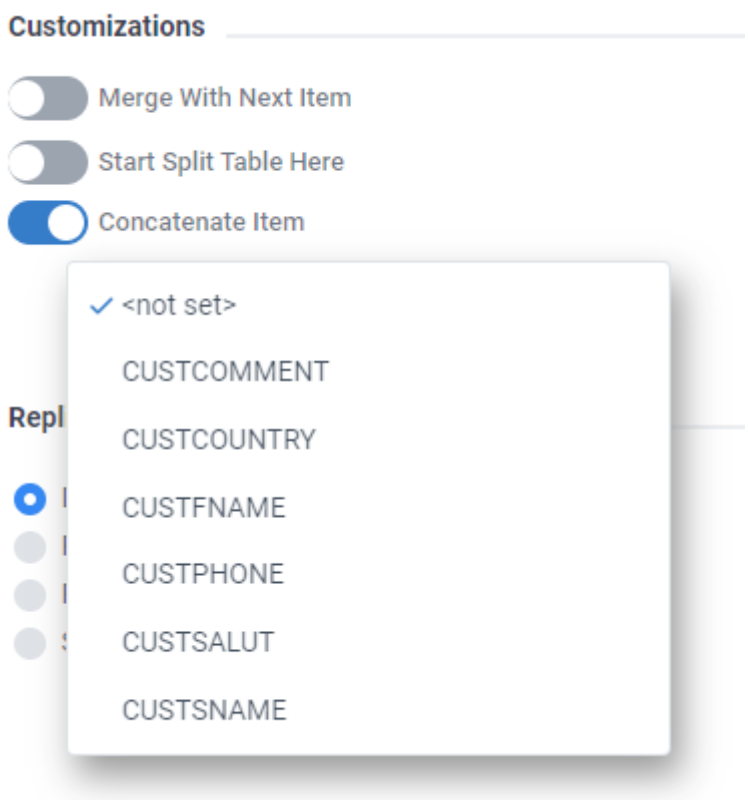
9.4.3 Merging Neighboring Items

Merging like items is a special case of concatenation this is performed at define/redefine time rather than at run time, and is therefore more efficient. The length of the fist item is extended to cover both items in the DATAITEMS table and the second item is not replicated to the relational database.

9.4.4 Concatenating Items

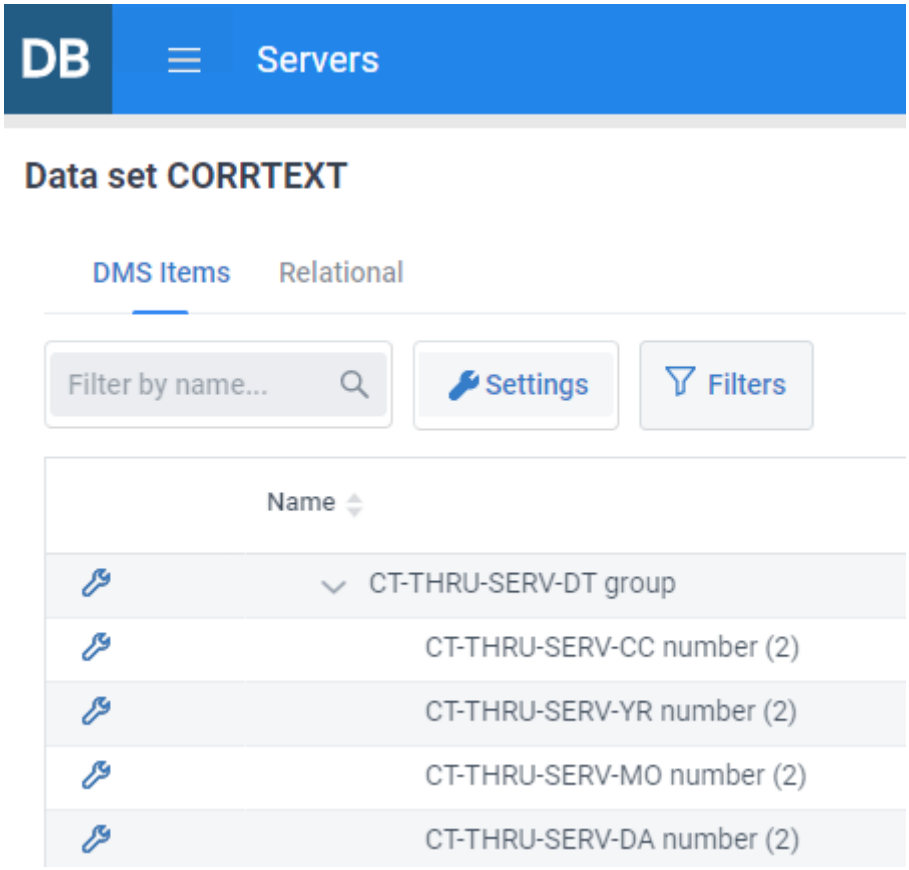
Concatenations are allowed between like items of type ALPHA or unsigned numbers (i.e. NUMBER(N)). Items customized as **Replicate as ALPHA** will be treated as if they were ALPHA(N) items in concatenations and items customized as **Replicate as Number** will be treated as if they were NUMBER(N) items.

The first step in setting up a concatenation is to enable the "Concatenate Item" option. This will expand a list box with the potential targets for the concatenation becoming visible. Select the item to combine with the current item to form a longer item. The most common type of concatenation is to concatenate a date (e.g. a NUMBER(8) that is date in the "YYYYMMDD" format) with time (e.g. a NUMBER(6) that is in the "HHMISS" format). The resulting NUMBER(14) can then be replicated as a date, which will be a timestamp (**datetime** or **datatime2** in the SQL Server Client and **date** in Oracle Client).

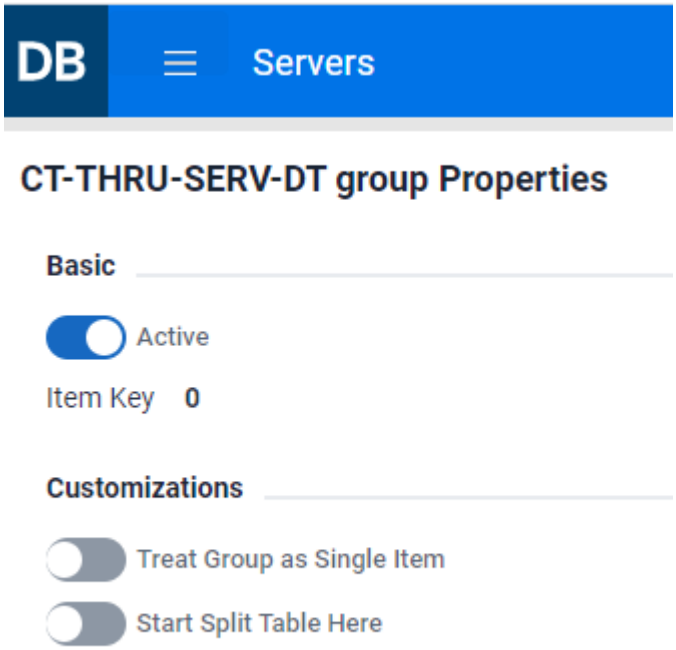


9.4.5 Redefining GROUPS of Like Items as Single Items

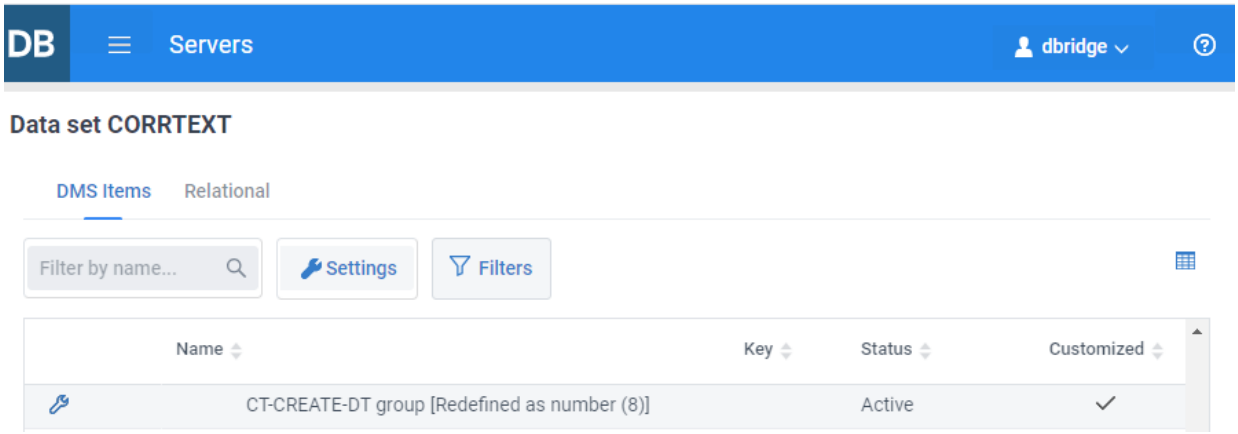
The items in the GROUP (or nested GROUPs) must all be either ALPHA(N) or unsigned numbers (i.e. NUMBER(N) items). When this option is enabled, the GROUP will be treated as a single item with the common data type whose length is the sum of length of all the items in the GROUP. For example:



The properties for this item look like this:



If you enable the option **Treat Group as Single item** and return to the DMS Item tab view the group will now look like a NUMBER(8) as shown below. This is particularly useful when you want to replicate the item as a date. If you go back into the redefined GROUP's properties you will now be able to replicate to a date.



9.4.6 Start a table split after the current item

This item, which corresponds to the DIOPT_Split_Table bit in the `di_options2` column of the DMS_ITEM Client control table, forces the **Customize** command to split the table before mapping this item. This gives the user more control in handling split tables when the splitting of the table happens in the middle of an OCCURS clause, which should be avoided.

9.4.7 End a split the table after the current item

This item, which corresponds to the DIOPT_End_Split_Table bit in the `di_options2` column of the DMS_ITEM Client control table, makes the Client return to the parent table following a forced split. It must follow an item with the DIOPT_Split_Table bit set and there can be only one outstanding split (i.e. you cannot have two table splits followed by two end table splits).

9.5 Customize DMS Settings

Use the **Customize** option to map individual DMSII data sets (and their items) and customize the table layout of the relational database.

Customizations should be made before cloning a data source or your changes may require that you re-clone. If you want to make customizations that affect the entire data source, use the **Configure** option available from the **Settings** drop-down button on the data sources page. See Customize the Client Configuration for more information on customizations for the entire data source.

> **Note**
>
> The customize command is a cooperative process between the **DBClntCfgServer** program, the Client Manager service, the code in the Administrative Console server, and the browser. Do not use the back button in the browser or close the browser before you successfully exit the **Customize** command by using the back button and **Done** button provided in the Administrative Console user interface. Closing the browser will leave the **DBClntCfgServer** program in a bad state. When this happens, the program can timeout, or the process can be ended by using the Task Manager.

When **Customize** is selected, the data set view will be displayed. In this view, you can customize data sets, deselect data sets you do not wish to replicate, set data set specific options that override global parameters specified in the Client configuration file, and link virtual and real data sets from which they are derived.

From the data sets view, select a data set to load the **DMS item** tab where customizations on the data sets DMS items can be performed.

DMS Item Customization Examples: - Allow you to disable items that you do not wish to replicate. - Modify how an item is replicated (e.g. a date represented as a NUMBER(8) in DMSII can be replicated as a relational database date) - Items with an OCCURS table can be mapped to a secondary table (or it can be flattened with the table).

Clicking on the **Relational** tab in the DMSII page takes you to the relational view where you customize data tables and data items. The **Done** button in both views takes you back to the data sets view.

Relational Customization Examples: - Rename data tables and data items. - Changing the order of columns in a table. - Modify the data types of an item. - SQL Server Client: - Apply data masking to sensitive items.

To exit the **Customize** command return to the data set view and push the **Done** button in the bottom-right of the Administrative Console.

9.5.1 DMS Items Page

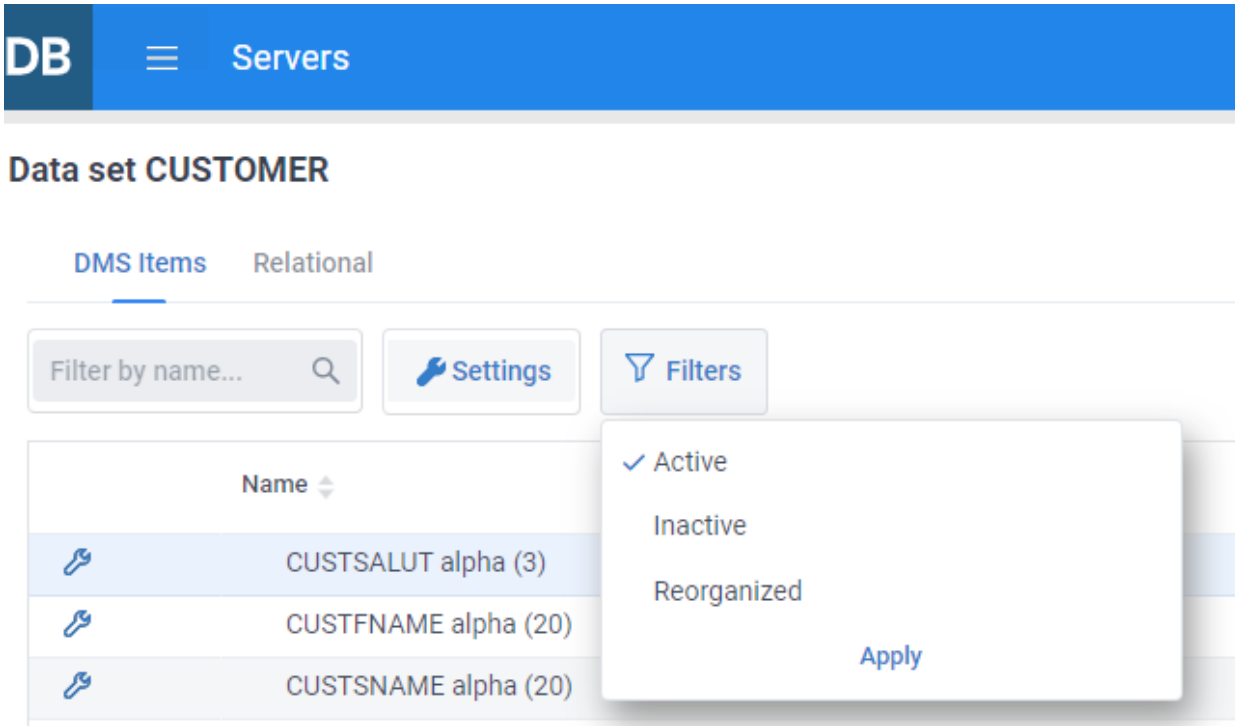
Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (select a data set)

Selecting a data set in the data sets view opens the **DMS items** tab for the data set. The **DMS items** tab is a table of DMS items which has 4 columns. The entries have a **Properties** icon to the left of the DMS item name that allows you to access the properties page for the item. This is followed by the "**Name**" column that contains the item name followed by the data type. The next column named "**Key**", contains a key icon for any item that is a member of the DMSII SET that is being used as the source of the index. The "**Status**" column reflects the value of the `active` column in the DMS_ITEMS Client control table, and the **Customized** column contains a checkmark when the value of a property for the DMS item has been changed.

> **Tip**
>
> Unlike the data sets view, the DMS items view can contain groups that need to be expanded to see the items.

Filters



Since several DMSII data sets typically have hundreds of items, you can filter items to make it easier to locate specific items. Typing a set of characters in the **Filter by name** edit box will only show you the data sets whose names contain the string of characters entered. The **Filters** button, when selected, will present you with 3 filtering choices, **Active** (default), **Inactive** and **Reorganized**. When the filter is set to **Active**, only DMS items whose `active` column is 1 in the DMS_ITEMS

Client control table will be shown. If you want to also see DMS items whose `active` column is 0, select **Inactive** in the **Filters** drop-down. Select **Apply** to view the DMSII items list results from the newly applied filter. The **Reorganized** option is used during the processing of DMSII reorganizations, where the view will be limited to data sets that have been affected by a reorganization. The Administrative Console will automatically enable this filter when it detects that it is processing a DMSII reorganization.

In addition to filtering, you can sort the table by **Name** (which is the default), **Key**, **Status**, or **Customized** by using the up and down arrows in the column headers.

> **Warning**
>
> The view that shows only what has changed from a DMSII reorganization will help prevent unnecessary changes to the layout of a dataset that was not affected by a reorganization.

> **Tip**
>
> For data sets that contain DMSII GROUP items the filtering will descend into the GROUPS, so if you have a naming convention ending the item name with "-DT" when they are dates, using this as a filter will find all the dates.

9.5.2 Customizing DMS Items

How to select a DMS Item

- Do one of the following:

  Select the properties icon  for the desired data item.

  -or-

  Click on the DMS item table row for the desired DMS Item (the table row will be highlighted in blue) indicating that it is selected and select **Properties** from the **Settings** drop-down button in the menu bar.

  Currently, there is not an **Information** page for DMS items, as all of the properties we display are editable.

Once a DMS item has been customized, return to the **Data Sets** page with the DMSII Items tab selected. Select the **Relational** tab to view the tables mapped from the data set in question and their columns. If the Client control tables have not yet been updated with the changes you made, pushing this button causes the data sets that need mapping to be mapped. You do not have to do this at this time; when you exit the **Customize** commend by navigating away from it using one of the provided links, this operation will be automatically performed. The Relational tab has a tab named **DMS Items** that gets you back to this page.

When you are done customizing the data set push the **Done** button to return to the data sets view. If you navigated to the relational view, you will be able to return to the DMSII item tab view or to the data sets view.

9.5.3 Relational Page

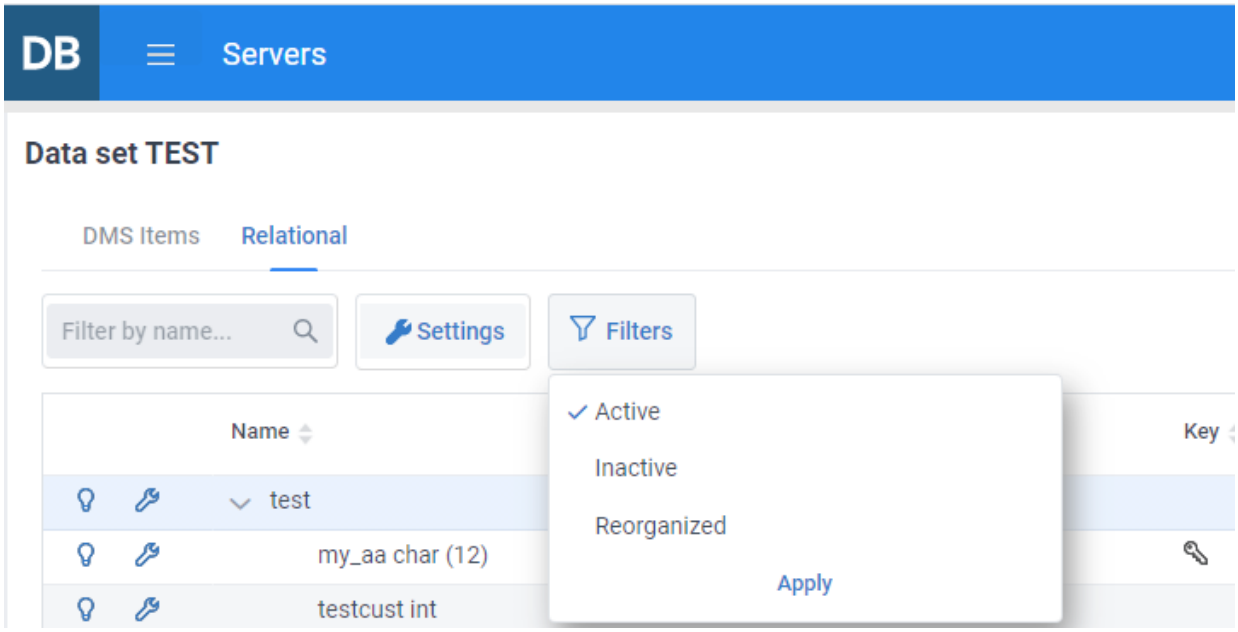
Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (click on data set) > **Relational**

Selecting the **Relational** tab in the DMS items view opens the Relational tab for the data set. The relational tab is a table of objects, which has 4 columns. The term "object" will be used to denote data tables and data items throughout this section. The entries have two icons to the left of the object name that allow you to access the **Properties** and **Information** pages for the object. This is followed by the **Name** column that contains table name or the column name followed by the data type. The **Key** column and contains a key icon for any item that is part of the index for the table it belongs to. The **Status** column reflects the value of the `active` column in the DATATABLES or DATAITEMS Client control table. Lastly, the **Customized** column contains a checkmark when the value of a property for the object has been changed.

Unlike the DMS items tab, the Relational tab is grouped by table, as there can be more than just one table associated with a data set.

Filters



Since several DMSII data sets typically have hundreds of items, you can filter items to make it easier to locate specific items. Typing a set of characters in the **Filter by name** edit box will only show you the data sets whose names contain the string of characters entered. The **Filters** button, when selected, will present you with 3 filtering choices, **Active** (default), **Inactive** and **Reorganized**.

When the filter is set to **Active**, only DMS items whose `active` column is 1 in the DMS_ITEMS Client control table will be shown. If you want to also see DMS items whose `active` column is 0, select **Inactive** in the **Filters** drop-down. Select **Apply** to view the DMSII items list results from the newly applied filter. The **Reorganized** option is used during the processing of DMSII reorganizations, where the view will be limited to data sets that have been affected by a reorganization. The Administrative Console will automatically enable this filter when it detects that it is processing a DMSII reorganization.

In addition to filtering, you can sort the table by **Name** (which is the default), **Key**, **Status**, or **Customized** by using the up and down arrows in the column headers.

> **Warning**
>
> The view that shows only what has changed from a DMSII reorganization will help prevent unnecessary changes to the layout of a dataset that was not affected by a reorganization.

> **Tip**
>
> For data sets that contain DMSII GROUP items the filtering will descend into the GROUPS, so if you have a naming convention ending the item name with "-DT" when they are dates, using this as a filter will find all the dates.

9.5.4 Customizing Data Tables and Data Items

How to select a Data Table or Data Item

- Do one of the following:

  Select the properties icon  for the desired data table or data item.

  -or-

  Click on the desired data item or data tables row (the table row will be highlighted in blue) indicating that it is selected. Select **Properties** from the **Settings** drop-down button on the menu bar.

Once you are done customizing an object, you will return to this page when you close the properties page by pushing one of the two buttons in the bottom-right corner of the page. Once you are done customizing the items in the relational view, you can select the **DMS Items** tab to return to the DMS items view, or you can select the **Done** button to return to the data sets view.

9.6 Data Items Properties

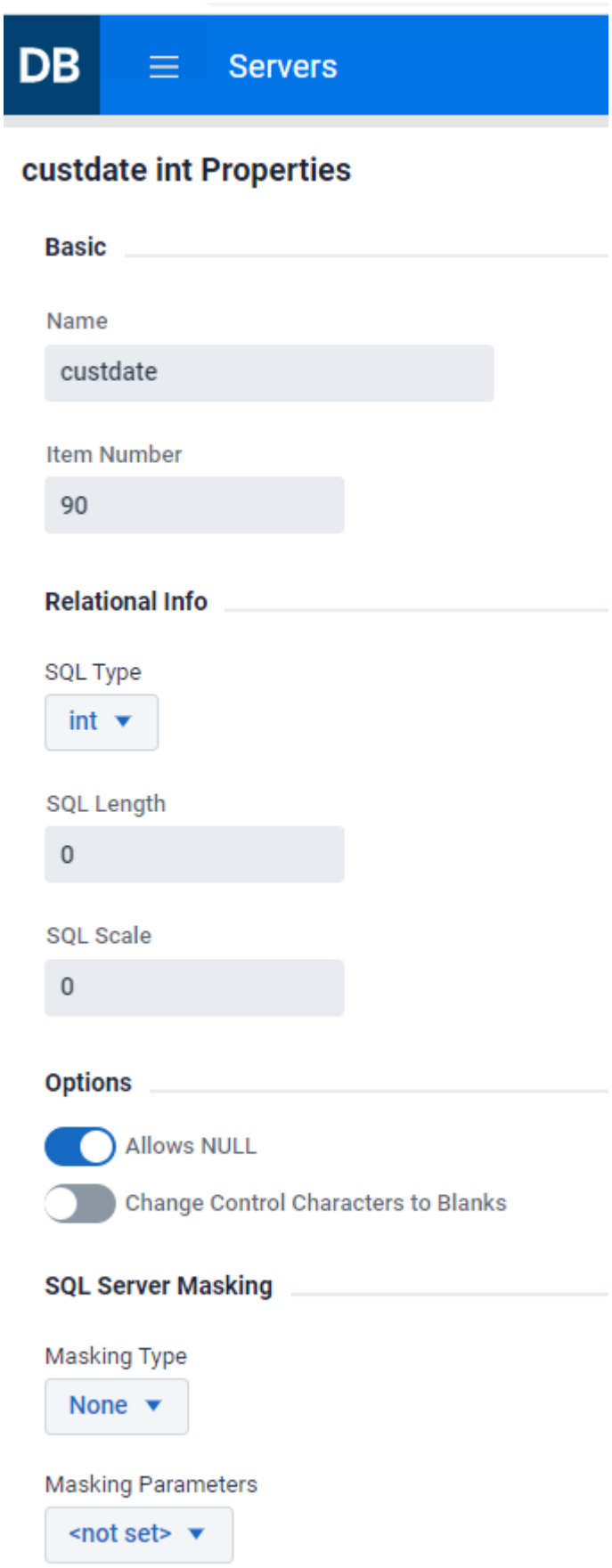
Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (select a data set) > **Relational tab** > (select a data item) > **Properties**

The customizable properties for a data item are organized in the following sections:

- Basic
- Relational info
- Options
- SQL Server Masking

These properties are dependent on the data type, the length of the item, and several other factors so the properties will vary from item to item, as we only show properties that are applicable to the specific item.


DB
Servers
custdate int Properties
Basic
Name
custdate
Item Number
90
Relational Info
SQL Type
int
SQL Length
0
SQL Scale
0
Options
Allows NULL
Change Control Characters to Blanks
SQL Server Masking
Masking Type
None
Masking Parameters
<not set>

9.6.1 Basic Properties

Name

Represents the value of the `item_name` column in the DATAITEMS Client control table for the item. You can rename an item by typing the new name in this input field. The name must conform to the database naming convention, cannot be a reserved word in the database, and also cannot be a duplicate of another column.

Item Number

This entry represents the value in the `item_number` column of the DATAITEMS Client Control table. It determines the position of the item in the DATAITEMS client control table. The "create table" statement used to create the table includes the column in this order. After a DMSII reorganization occurs, this may not match what is actually in the table, as new columns get added using an "alter table add column" statement that adds them to the end of the table. The order of items will affect stored procedures, as the order of the parameters must match what is in the control tables. If you renumber items you will need to run a **Generate Scripts** and a **Refresh** command for the data set to remedy this situation. The client number columns are in increments of 10 to make it easier to rearrange them by modifying the item number.

> **Tip**
>
> Do not renumber key items, as the client expects them to be at the start of the table.

9.6.2 Relational Info

This section contains information about the SQL type. The SQL type can be changed by using one of the options in the **SQL type** list box by selecting the down arrow. Select the desired item from the list. If the selected SQL type has a length or scale, the data type must have a value of greater than equal to the original value, as specifying a smaller value would result in data truncation.

9.6.3 Options

This section has two configurable properties as seen below:

Allow Nulls

This property gets its initial value from the configuration parameter `allow_nulls`, which only applies to items that are not keys. It corresponds to the DAOPT_Nulls_Allowed bit the `da_options` column of the DATAITEMS table entry. You can change this bit as long at the item is not a key. In the case of a MISER database where the, unless the parameter `use_nullable_dates` is set to True, keys that are MISER dates will allow nulls.

Change Control Characters to Blanks

This property gets its initial value from the configuration parameter `convert_ctrl_char`. It corresponds to the bit DAOPT_FixAlphaChar bit in the `da_options` column of the DATAITEMS table entry.

9.6.4 SQL Server Masking

This property, which only applies to the SQL Server client, allows you to apply data masking to the column by adding the appropriate specification to the column defintion in the DDL. The Databridge Administrative Console breaks this down into two components, which are the **Masking Type** and the **Masking Parameters**

Masking Type

The following masking types are defined:

- no masking
- default
- email
- random
- partial

These are presented using a list-box from which you can select the desired type. The last two entries have parameters which are supplied in the **Masking Parameters** list box. The random function applies to numeric items, it has 2 parameters, which are the **minimum** and **maximum** value for the random numbers that will be displayed instead of the actual value for the column. The partial function applies to character data and has 3 parameters, the number of characters to show at the start of the data, the mask for intermediate characters in the data, and the number of characters at the end of the data that is shown. For example, you could enter a 0 character at the start and 4 at the end to show the rest of the data as asterisks.

Masking Parameters

The Administrative Console implements the parameters as an array of strings whose entries are defined in the configuration file and associates with an index value in the range 1 to 100. The `masking_info` column of DATAITEMS, which defines the masking function and the index of the corresponding parameter string, which does not include the parentheses. The format of the `masking_info` column (which is an int) is 0x00*nn*000*m*, where *m* is the masking function code and *nn* is the index into the masking table.

An example for this is a value of `0x00010003` for `masking_info`, which represents a masking type of 3, which is random masking, with its parameters represented by the `masking_parameter[1]` entry in the configuration file. This parameter could be `"0,100"` which would result in the masking function `"random(1,100)"` being used in defining the data mask for the column.

You can reuse `masking_parameter` entries as many times as needed. The index must be between 1 and 100. Refer to the SQL Server documentation for details on how data masking works.

This information is used to make the line for the column `accountno` in the DDL looks this this:

```
accountno varchar(16) masked with (function='partial(0,"************",4)') NULL,
```

9.7 Data Item Information (Read-Only Properties)

Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (select a data set) > **Relational** > **Settings** > (select a data item) > **Information**

The read-only properties for a data item are organized into **Basic** and **Options** sections.

9.7.1 Basic Properties

This section has one updatable property, **Table Name**.

Data Type

This entry is redundant as the data type is already included in the item name.

Derived from

This entry displays the name and data type of the DMS item from which the item was derived.

Active

This entry shows the value of the `active` column for the corresponding entry in the DATAITEMS Client control table.

Item Key

This entry corresponds to the `item_key` column in the DATAITEMS Client control table for the item.

9.7.2 Options

The entries below can be useful to find out why the column is marked as being customized.

Column Renamed

Indicates that the column was renamed.

Length Changed

Indicates that `sql_length` was changed in the corresponding DATAITEMS entry.

Scale Changed

Indicates that `sql_scale` was changed in the corresponding DATAITEMS entry.

Replicating as Char

Indicates that the `da_options` bit DAOPT_Store_as_Char is set in the corresponding DATAITEMS entry. This is the result of a customization done in the DMSII view that enables the **Replicate as Alpha** option.

Translating Binary Data

Indicates that the `da_options` bit DAOPT_Xlate_Binary is set in the corresponding DATAITEMS entry. This is the result of a customization done in the DMSII view that enables the **Replicate as Binary** option and sets **Translate Binary Data** to True.

Replicating as Number

Indicates that the `da_options` bit DAOPT_Store_as_Number is set in the corresponding DATAITEMS entry. This is the result of a customization done in the DMSII view that enables the **Replicate as Number** option.

Type Changed

Indicates that `sql_type` was changed in the corresponding DATAITEMS entry.

Using Custom Date Format

Indicates that the `da_options` bit DAOPT_VarFormat_Date is set in the corresponding DATAITEMS entry. This bit tells the Client how to interpret the `dms_subtype` column. This is the result of a customization done in the DMSII view that enables the **Replicate as Custom Date** option.

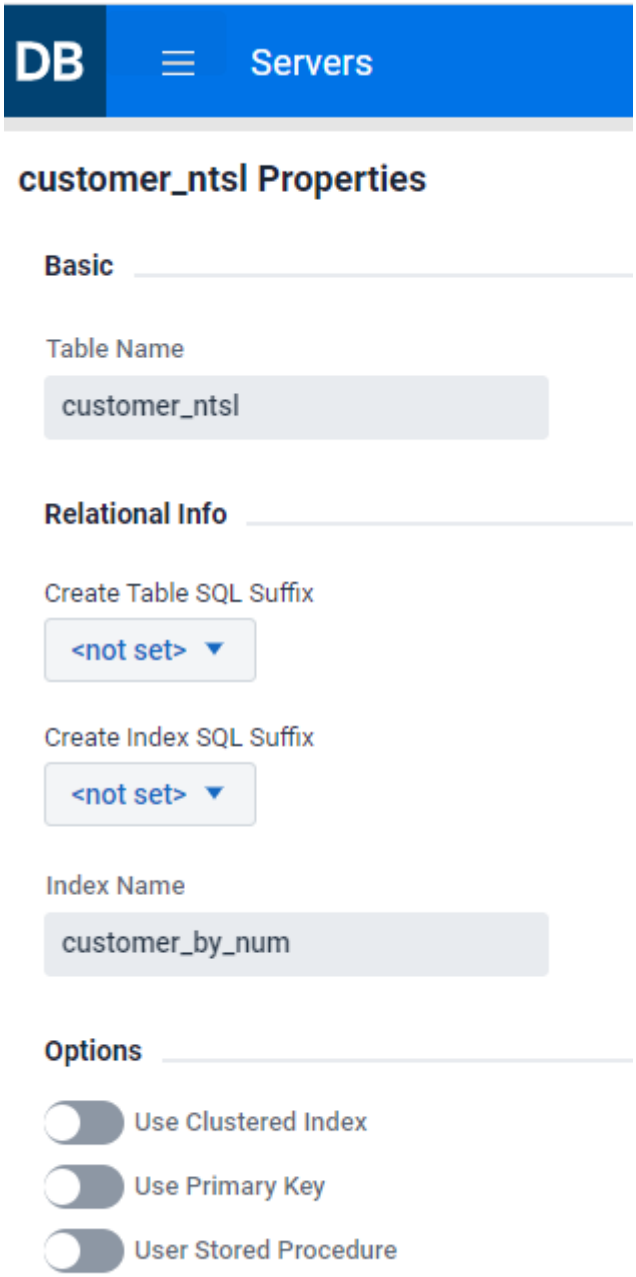
9.8 Data Table Properties

Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (click on data set) > **Relational** > **Settings** > **Properties**

The customizable properties for a data table are organized in the following sections:

- Basic
- Relational Info
- Options



9.8.1 Basic Properties

This section has one updatable properties, **Table Name**.

Table Name

This entry represents the value of the `table_name` column in the DATATABLES Client control table for the item. You can rename a table by typing the new name in this input field. You must use a name that conforms to the database naming conventionn while also avoiding names that are reserved words in the database or that duplicate other table names in the database. When a table is renamed, all the entries in the DATAITEMS need to be changed to have the new table. This process is handled automatically by the Administrative Console to simplify the process of renaming a table.

9.8.2 Relational Info

This section has two entries **Create Table SQL Suffix** and **Create Index SQL Suffix**. The entries allow you to add optional extensions to the "create table" and "create index" SQL statements for the table. Use these entries to control how the table allocates more space when it is full, or TABLESPACE where the table is to be created (TABLESPACE in the case of Oracle and filegroup in the case of SQL Server).

The suffixes are defined in the Client configuration file using the `create_table_suffix[n]` and `create_index_suffix[n]` parameters. These parameters allow you to pick a string from the corresponding list and set the `create_suffix` or `index_suffix` columns in the DATATABLES Client control tables to be the index of the selected string. Refer to the **Configure** command in the data source's **Settings** menu to find out more about SQL suffixes.

9.8.3 Options

This section has three editable properties **Use Clustered Index**, **Use Primary Key** and **User Stored Procedure**.

Use Clustered Index

(SQL Server client only) This option corresponds to the `dt_options` bit DTOPT_Clustered_Index in the DATATABLES Client control table, which gets its initial value from the parameter `use_clustered_index`. See the *Databridge Client Administrator's Guide* for details on the various index types.

Use Primary Key

This option corresponds to the `dt_options` bit DTOPT_PrimaryKey in the DATATABLES Client control table, which gets its initial value from the parameter `use_primary_key`. See the *Databridge Client Administrator's Guide* for details on the various index types.

User Stored Procedure

This option only applies to MISER databases. It is used to make the client run the stored procedure *m_tablename* to process a CREATE record for the table, instead of handling it like a normal CREATE record.
This option corresponds to the bit DTOPT_UserSP in the `dt_options` columns of the DATATABLES Client control table.

9.9 Data Table Information (Read-Only Properties)

Getting there

Databridge Servers > **Client Managers** > **Settings** > **Customize** > (click on data set) > **Relational** > **Settings** > **Information**

The read-only properties for a data item are presented as Basic, Relational Info, and Options.

9.9.1 Basic Properties

Dataset Name

This entry represents the value of the `dataset_name` column in the DATATABLES Client control table for the item.

Active

This entry represents the value of the `active` column in the DATATABLES Client control table for the item. This will normally be set to True, the only exceptions being items that are the tail ends of concatenations will have **Active** = False. The item must be present during a `process` command to extract the data for the concatenation.

9.9.2 Relational Info

This section has a single entry. **Primary Table** indicates whether or not the table is a primary table. Secondary tables are tables created for items where OCCURS clauses are not flattened or they are the result of a table being split into multiple tables because of restrictions in the relational database (e.g. limit on the maximum number or columns).

9.9.3 Options

This section primarily has one property. The remaining read-only properties are only shown when they are set to True.

History Table

Indicates whether or not the table is a history table. A **History Table** is a special table that maintains a record for every update.

Index Renamed by User

Indicates that the `index_name` column in the corresponding DATATABLES entry was modified.

Preserving Deleted Records

Indicates that the table is preserving deleted records in one of two ways. The first method is to use the user column (expanded) `update type`, where deleted records are left in the table after setting the update type to **DELETE (2)**. The second method, adds the user column `deleted_record` and optionally the column `delete_seqno`, where a delete puts the current date/time into the `deleted_record` column and a sequence number to prevent duplicate records that occur when the record is deleted, reinserted, and deleted again within the same second. This can happen with applications that update by deleting and reinserting records.

Table Contains Only Keys

This indicates the all the columns in the table are keys.

Table has OCCURS DEPENDING ON

Indicates that the table is an occurs table that is derived from an item with an OCCURS DEPENDING ON clause. Such tables need special processing, as the number of occurrences can change, and we only store as many occurrences as the item in the OCCURS DEPENDING ON clause specifies.

Table Renamed by User

Indicates that the `index_name` column in the corresponding DATATABLES entry was modified. When a table is renamed, the `table_name` columns of its data items in the DATAITEMS table are also renamed.

Table Split

Indicates that this table is part of a split, where the item in the data set ends up in multiple tables because the table has too may columns, or, the user decided to split the table.

10. Global Parameters

10.1 Global Parameters

You can set global parameters for the following:

- Bulk loader
- Customizing
- Logging
- Processing
- PCSpan
- Encryption

10.2 Bulk Loader

Use the following parameters to control the bulk loader (bcp or SQL*Loader) utility for your relational database (SQL Server or Oracle). Use the following parameters to customize the selected data source before cloning.

Configuration file parameters are included below with the `following_font`.

USE BCP UTILITY TO LOAD TABLES

Parameter: `use_bcp`

(SQL Server only) The SQL server Client can operate with the bcp utility or the BCP API. This parameter determines the default value used by the `define` and `redefine` commands when setting the `ds_options` for the various data sets. It is recommended to use the bcp utility as it is more reliable than the BCP API.

DELIMITER (5 CHARACTERS OR LESS)

Parameter: `bcp_delim`

(SQL Server only) Specify the character or set of characters that are used to separate variable-length fields in the bulk loader input records.

Enter one or more characters. Use this option if the data contains alpha fields with TAB characters that need to be preserved. (A possible delimiter value in this case would be "|" or "||".)

CODE PAGE (5 CHARACTERS OR LESS)

Parameter: `bcp_code_page`

For SQL Server, you can use the names ACP, EOM, RAW or a number such as 1252 to represent a code page. For Oracle, the name is a string such as "WE8ISO8859P1". Consult your Oracle documentation for the actual names.

COPIED MESSAGE (32 CHARACTERS OR LESS)

Parameter: `bcp_copied_msg`

(SQL Server only) Allows the bcp_auditor utility to determine whether or not a bulk loader was successful in cases where the database language is not English. For example, in German, this parameter is "Zeilen kopiert", but in English, it is "rows copied" (default). If this parameter is not set correctly, the bcp_utility reports bulk loader failures even though the bulk loader worked correctly.

BATCH SIZE (1000 MINIMUM TO 10000000 MAXIMUM)

Paramter: `bcp_batch_size`

(SQL Server only) Specifies the bcp utility batch size, which is the number of rows per batch of data copied. This allows the bcp utility to load a table in several batches instead of in a single operation. Permitted values are 0 or 1000-10000000 (rows per batch). A value of zero causes the bcp utility to load the entire group of records for the data file in one batch. Copying all of the rows of a very large table in one batch may require a high number of locks on the Microsoft SQL Server database.

PACKET SIZE (512 MINIMUM TO 65535 MAXIMUM)

Parameter: `bcp_packet_size`

(SQL Server only) Defines the network packet size value for the bcp utility (applies to remote servers only). If you have wide tables, setting this parameter to a packet size larger than the default (4096) can speed up loading the data into the table at the expense of system resources. Before you can set the value for this parameter, you must first enable the option in the Administrative Console.

SQL LOADER NUMERIC CHARACTER

Parameter: `bcp_decimal_char`

(Oracle only) This parameter is normally auto-detected by the Client and gets its value by reading the value of Oracle database's NLS_NUMERIC_CHARACTERS parameter.

This method will work correctly when the Client and the database reside in the same machine. However, if the Client is run outside the database machine, it is not guaranteed the the Oracle Client software used by the Databridge Client will have the same NLS settings as the target database.

For example, it is possible to have a US Oracle Client in the Client machine that connects to a Brazilian database. In this rather unusual situation, you would have to set the SQL loader numeric character to `'.'` as it will default to `','` which would lead to SQL*Loader errors for all records that have numeric data with a decimal point.

SQL LOADER PARALLEL

Parameter: `enable_parallel_mode`

(Oracle only) This parameter, which is only meaningful when DIRECT mode is enabled, causes the **Generate** command to add the specification parallel = True to the SQL*Loader command line. Parallel mode makes the SQL*Loader run faster at the expense of additional system resources.

SQL DIRECT

Parameter: `inhibit_direct_mode`

(Oracle only) Controls whether the **Generate** command adds the specification direct=True to the SQL*Loader command line. If your Oracle database is on the same machine as the Databridge Client, you would let this parameter assume its default value of False, as DIRECT mode is much faster than CONVENTIONAL mode. Conversely, if your Databridge Client accesses a remote Oracle database using SQL*Net between two dissimilar architectures (for example, Windows and UNIX), you must use the CONVENTIONAL mode by setting this parameter to True.

Setting **SQL direct** to False inhibits the use of the "direct=True" option when invoking SQL*Loader in the command files. When you set **SQL direct** to False, it is recommended to increase the size of **SQL loader bind size** for better performance.

SQL LOADER BIND SIZE

Parameter: `sqlld_bindsize`

(Oracle only) Defines the value to be used for the BINDSIZE parameter for SQL*Loader operations. Increasing this value can speed up SQL*Loader operations when using conventional mode (for example, running remote to a database on a UNIX system). Use **SQL loader rows** and **SQL loader bind size** when you are running the Databridge Client for a remote Oracle database running on UNIX or Windows. A larger bind size and row size can increase the speed of the load across Oracle Network Services at the expense of using more memory.

SQL LOADER ROWS

Parameter: `sqlld_rows`

(Oracle only) Defines the value to be used for the ROWS specification for SQL*Loader operations. Use **SQL loader rows** and **SQL loader bind size** when you are running the Databridge Client for a remote Oracle database running on UNIX or Windows. A larger bind size and row size can increase the speed of the load across Oracle Network Services at the expense of using more memory.

MAXIMUM TEMPORARY STORAGE (40M MINIMUM TO 3G MAXIMUM)

Parameter: `max_temp_storage`

(Windows only) Specify a value between 10MB and 3GB for the maximum amount of storage the Databridge Client will use for temporary files. Use the letter **M** or **G** after the number to indicate megabytes or gigabytes.

MAXIMUM LOADER ERRORS (0 MINIMUM TO 1000 MAXIMUM)

Parameter: `max_errors`

Controls the bulk loader's tolerance to records that are discarded due to data errors. Increasing the maximum error count allows you to gather all the errors in one run rather than finding 10 errors and then having to start over again.

MAXIMUM NUMBER OF FAILED LOADS (0 MINIMUM TO 25 MAXIMUM)

Parameter: `max_bcp_failures`

Specifies how many failed loads the Client will allow before aborting the run.

VERIFY LOAD

Parameter: `verify_bulk_load`

Specifies how the Databridge Client will handle the results of the bulk copies.

When **Do not verify** is selected, no action is taken. When **Verify** (the default) is selected, the Databridge Client gets the record count of each table and compares it to the total number of records passed to the bulk loader after the final bulk load is completed. If the two counts differ, the Databridge Client displays a warning message. **Verify, exit on error** is the same as Verify except that the Client terminates so that you can investigate the reason for the mismatch.

10.3 Customizing

10.3.1 Customizing

You can customize parameters for general or specific settings.

- General
- Advanced
- History Tables
- SQL Data Types
- SQL Suffixes
- Translations
- User Columns Section One
- User Columns Section Two
- User Scripts

10.3.2 Customizing - General

Use the following parameters to customize the selected data source before you clone it.

Configuration file parameters are included below with the `following_font`.

General

Allow NULLs

Parameter: `allow_nulls`

Defines the default way of handling non-key items that are NULL in DMSII or contain bad values. When this parameter is set to True, the data item's `da_options` column will have its DAOPT_Nulls_Allowed bit set to True by **Define/Redefine** and **Customize** commands. This will result in the column having the NULL attribute. Conversely, if the parameter is set to False, the columns will have the NOT NULL attribute. In this case null data will be stored using specific values designated to represent nulls. Null character data will be stored as blanks. Null numeric data will be stored as either 0 or all nines based on the value of the `null_digit_value` parameter. Null dates will be stored using dates like 1/1/0001 (Oracle) or 1/1/1900 (SQL Server).

Ignore REQUIRED attribute

Parameter: `inhibit_required_opt`

This option instructs the Client to ignore the REQUIRED attribute of DMS items. The Client (version 6.5 and newer) honors the VALUE REQUIRED property for items that are members of SETS where NULL is not allowed. The Client uses this information to make such column(s) not allow nulls. Older Clients, which did not have access to this property, allowed nulls for such items when the **Allow Nulls** parameter was set to True. To ensure backward compatibility, the Client ignored the REQUIRED attribute for data sources created by older versions of the software. This prevented such column(s) from having their NULL attribute changed to NOT NULL when a **Redefine** command was run.

This parameter makes new data sources operate in the same way as they did with older Clients, which did not have access to the REQUIRED attribute. This is particularly useful when a date value of 0 represents a NULL date in DMSII, even though the item is not NULL in DMSII. This parameter enables the Client to store NULL dates and dates that have bad values as NULL, rather than using an artificial date to represent NULL dates, when applications expect this behavior.

Set blank columns to NULL

Parameter: `set_blanks_to_null`

This parameter indicates that the Client will store zero-length character data (that is, "") as NULL instead of a single space. This parameter only applies to columns that are not part of the index.

Allow NULL dates

`use_nullable_dates`

Allows the Client to treat a single MISER date in the index as nullable.

Use brackets around tables names that are reserved words

Parameter: `bracket_tabnames`

Allows the SQL Server Client to use table names that are reserved words by enclosing them in square brackets.

Ignore new data sets

Parameter: `suppress_new_datasets`

Indicates whether or not the Client will inhibit new data sets created during a DMSII reorganization from being mapped to relational database tables. If this parameter is set to True, the `active` columns for new data sets are set to 0. If you later decide that you want to replicate these data sets, enable the **Active** checkbox in the data sets properties page and then run a **Redefine** or a **Customize** command.

> **Caution**
>
> Do not mix **Redefine** and **Customize** commands, as the **Redefine** command will not work when you first run a **Customize** command. Using a **Redefine with options** command and specifying the `-u` option will allow the command to run, however, any customizations from the **Customize** command will be lost.

Use column prefixes

Parameter: `use_column_prefixes`

Extends the table name prefix specified in the DATASOURCES Client control table to all column names.

If a prefix is not defined, this setting has no effect.

Strip Data Set prefixes

Parameter: `strip_ds_prefixes`

This parameter makes the **Define/Redefine** and the **Customize** commands set the `item_name_prefix` column in the DATASETS table to the data set name. This is useful when all DMSII data item names use a common prefix; for example, using the data set name followed by a

dash. This parameter provides a quick way of stripping those common prefixes without writing any user scripts or using the **Customize** command (as renaming every column requires a lot of work). If the prefix is an abbreviated form of the data set name (e.g. SVHIST instead of SV-HISTORY), start the **Customize** command and select the data set in data sets view of the command, then open its properties page by clicking on its properties icon  . Then set the **Item Name Prefix** property to this value.

DMSII related parameters

Enable DMSII links

Parameter: `enable_dms_links`

Allows Databridge to implement DMSII links. DMSII links are implemented using AA values as foreign keys. To use this parameter, you must set the Links parameter to True in the DBEngine control file. For more information, see the *Databridge Host Administrator's Guide*.

Track Variable Format data sets that have LINKS

Parameter: `track_vfds_nolinks`

Allows the Client to track variable-format data sets that contain links. The links themselves are not tracked or updated. When a record is created in a variable-format data set, links are set to NULL. If the application assigns the links to point to other records, the Client database will not contain these new link values until the variable-format data set is re-cloned. This parameter is enabled by default.

When this parameter is disabled, variable-format data sets are set to mode 11 after an initial cloning and do not get updated.

Propagate Type 0 changes for VF data sets

Parameter: `global_type0_changes`

The **DBClntCfgServer** program and the Administrative Console apply all customizations done to the fix part of a variable format data set to all the records types, as they all contain the exact same fixed part. Some sites have variable format data sets that have a large number of record types, if you are customizing a date in the fixed part using the **Customize** command, you only have to do this once and it gets applied to all the records types. This parameter is provided to allow users to disable this feature.

Extract embedded data sets

Parameter: `extract_embedded`

Embedded data sets are not normally handled by the Databridge Engine when INDEPENDENTTRANS is False, as the Client cannot apply fixups or updates. When this parameter

is enabled, the Client enables a DBEngine parameter that causes it to go through the data extraction process for embedded data sets that cannot be tracked. When this parameter is disabled, Databridge Client ignores embedded data sets.

If INDEPENDENTTRANS is True, the Databridge Client can clone and update embedded data sets regardless of the setting of this parameter.

Read NULL record values

Parameter: `read_null_records`

Causes the **Define/Redefine** and **Customize** commands to get NULL VALUES for data set records from DBEngine. NULL VALUES are stored in a binary file (`datasource_NullRec.dat`) and are retrieved at the beginning of a `process` or a `clone` command. If this parameter is disabled, numeric data items whose bits are all 1 (high values) and character data items whose bits are all 0 (low values) are considered to be NULL. When this parameter is enabled, testing for NULL is more accurate.

MISER database

Parameter: `miser_database`

Enable this parameter when running Databridge at a MISER database site. When enabled, this parameter sets the default date format to a MISER date and additional parameters that are required for a MISER site (if not already set): Flatten all OCCURS; Automated virtual data sets; Allow Null dates.

Preserve DMSII MASKING option

Parameter: `auto_mask_columns`

Directs the SQL Server Client to use the default data masking option for all columns whose corresponding DMSII item is masked in DMSII.

AA Values and RSN's

Use the corresponding buttons to specify how to represent absolute address (AA) values, record serial numbers (RSNs) and DMSII links. RSNs are unique serial numbers that get assigned when records get created and remain associated for the life of the record. RSNs must explicitly be enabled in the DASDL.

Use Binary AA Values

Parameter: `use_binary_aa`

Indicates that AA values (including RSNs, visible RSNs, Parent_AA values and DMSII links) will be stored as binary data, which reduces the storage requirement. Instead of using 12 bytes, which is

needed by character data, these values are stored using 6 bytes: binary(6) for SQL Server, or, raw(6) for Oracle.

Use character AA Values

Parameter: N/A (both `use_binary_aa` and `use_decimal_aa` are False)

Indicates that AA Values, Parent_AA values, RSNs, visible RSNs and DMSII links will be stored as CHAR(12), where each character is the hexadecimal representation of the corresponding digit (half-byte) in the A-Series word.

Use decimal AA Values

Parameter: `use_decimal_aa`

Indicates that AA values, Parent_AA values, RSNs, visible RSNs, and DMSII links will be stored using a numeric data type instead of char(12). The data type varies from database to database. For SQL Server, depending on the setting of the `use_bigint` parameter, either BIGINT or DECIMAL(15) is used. For Oracle, NUMBER(15) is used.

10.3.3 Customizing - Advanced

Use the following parameters to customize the selected data source before being cloned.

Configuration file parameters are included below with the `following_font`.

> **Note**
>
> Changing the parameters described in this page requires that you run a **Customize** or **Redefine** command to propagate the changes to the corresponding bits in the Client control tables various options columns. If you use a **Redefine** command you need to use **Redefine with options** from the **Advanced** button on the data sources page and enable the **Redefine all data sets (-R)** option.

Global data set options

Clear duplicate records encountered during data extraction

Parameter: `clr_dup_extr_recs`

This parameters defines the initial value for the data set property that causes the Client to remove false duplicate records after the index creation fails. The duplicate records are caused by long cloning of an active DMSII database. Clearing the duplicate records allows the index creation and fixup phases to continue. If false duplicate records aren't manually removed, index creation will fail. Running a **Customize** or **Redefine with options** command sets the `ds_options` bit that corresponds to the data set property **Clear Duplicate Extract Records** for all DATASETS table entries. Using the **Customize** command allows this option to control individual data sets.

> **Caution**
>
> Using this option is not recommended when you created a composite key or a GenFormat primary key, as the keys you created might result in duplicate records. Running the script in this case will get rid of all the duplicate records and force you to re-clone to recover the duplicate records. It is recommended to find out what the duplicate records are and determine if your index is good before running this script.

Force AA Values as indexes

Parameter: `force_aa_value_only`

This set of buttons allows you to define the default value of the property that makes datasets use the AA values or record serial numbers (RSNs) as the source of the index instead of the SET picked by the Databridge Engine. See below for an explanation of the available options.

- **Do not force:** the **Define/Redefine** and **Customize** commands will honor the Databridge Engine's decision on whether or not to use AA Values.
- **Always force:** forces the **Define/Redefine** and **Customize** commands to use AA values as the index, even if the data set has a SET that qualifies for use as an index.
- **Force only if RSN:** forces the **Define/Redefine** and **Customize** commands to use RSN values (when they exist) as the index.

Ignore new columns

Parameter: `suppress_new_columns`

When this parameter is set to True, the **Define/Redefine** and **Customize** commands set the `active` column to 0 for new DATAITEMS and DATATABLES entries that are associated with the reorganization. If you later decide that you want to include such columns, you must disable this parameter and run a **Redefine** command by clicking **Redefine (with options)** from the **Advanced** button on the data sources page and enable the **Redefine All Data Sets (-R)** option, unless you are using the **Customize** command.

Update changed columns only

Parameter: `minimize_col_updates`

This option specifies whether the **Define/Redefine** and **Customize** commands should set the `ds_options` bit that corresponds to the data set property **Update changed columns only** for all DATASETS Client control entries. This bit indicates that Client will only update columns whose values have changed. To do this, stored procedures are abandoned in favor of pure SQL without the use of host variables. This usually slows down the update speed of the Client considerably. However, when using SQL Server or Oracle replication, the overall process ultimately takes less time because significantly less data is sent to the remote database during the replication.

> ⚠ **Caution**
>
> Using this parameter will significantly slow update processing by the Client. If you are replicating your relational database, enabling this feature may improve performance if replication is very slow. Do not set this parameter to True under any other circumstances.

Optimize SQL updates

Parameter: `optimize_updates`

This option eliminates all redundant updates. Use this option when there is a large number of occurrences for items when OCCURs clauses are not flattened. An OCCURS clause is a DMSII construct that describes the number of times an item is present in a data set. This parameter globally sets the `ds_options` bit that corresponds to the data set property **Optimize SQL Updates** for all DATASETS table entries.

Split variable format data sets

Parameter: `split_varfmt_dataset`

The Client normally stores each record type in a table whose name is the data set name followed by "_type*nnn*" (where *nnn* is the variable format record type). The table for type 0 records, which have no variable part, does not have a suffix like the tables for all other record types. All record types start off with the fixed part which is followed by the variable part, except the type 0 records that have no variable part.

This option provides an alternate way for mapping variable format data sets to tables in the relational database. The fixed parts of all variable format records are stored in the table for type 0 records. The keys and variable parts of all the remaining record types are stored in the corresponding tables.

This option globally sets the `ds_options` bit that corresponds to the data set property **Split variable format data sets** for all DATASETS Client control table entries.

Use stored procedures in updates

Parameter: `use_stored_procs`

This parameter makes the `process` and `clone` commands generate the actual SQL command instead of a stored procedure call to perform an update. The Client still uses host variables, as was the case with stored procedure calls. Executing the SQL directly eliminates some overhead and makes processing the update quicker.

After executing a **Redefine** or **Customize** command, the Administrative Console will ask you to run a **Reorganize** command, which generates a new set of scripts for creating the tables. It also refreshes the stored procedures for all data sets by dropping them if no longer needed, or by recreating them if needed.

Table layout

Flatten all OCCURS

Parameter: `flatten_all_occurs`

Defines the initial value for the `di_options` bit in the DMS_ITEMS table entries that corresponds to the **Flatten all OCCURS** property..

This option creates a new column in the primary table for each OCCURS item. Enable this option if the DMSII data contains a lot of OCCURS clauses that you want to flatten. When this parameter is disabled, a DMSII data set that has an OCCURS clause in an item will be placed in a secondary table in the relational database. As a result, a single DMSII update can lead to updating multiple tables multiple times, as each occurrence of the item is placed in a separate row in the secondary table.

Maximum columns in tables (1 minimum to 1024 maximum)

Parameter: `maximum_columns`

Use this option to limit the maximum number of columns that can be created in split tables. Split tables are created from a data set that exceeds the maximum number of columns allowed by the relational database. For SQL Server, this number is 1024. If you set this parameter to 1000, the split will occur after 1000 columns.

Indexes

Use clustered indexes

Parameter: `use_clustered_index`

(SQL Server only) Enable this option to use clustered indexes for all tables. To override this option for a single table, disable the Use Clustered Index check box in the Relational Properties page for the table when using the **Customize** command.

Use Primary Keys

Parameter: `use_primary_key`

This option creates a primary key instead of using a unique index for all tables. To override this option for a single table, disable the **Use Primary Keys** option in the Relational Properties page for the table when using the **Customize** command.

Table reorganization options

Use internal clone for reorganizations

Parameter: `use_internal_clone`

This option affects the **Redefine**, **Customize** and **Reorganize** commands. Instead of using ALTER commands to add, delete or modify columns in tables, the Client uses a combination of scripts and table renaming commands to create new copies of the tables with new layouts. The Client then copies the data using SELECT INTO for SQL Server and CTAS (Create Table As Select) for Oracle.

This operation works like the bulk loader and is faster than using ALTER and UPDATE commands on large tables, but more importantly, the command is not logged.

> **Note**
>
> The only drawback to this method is that it requires increased free disk storage to hold a second copy of the table for the duration of the operation. This method is recommended over re-cloning.

Reorg command update batch size (5000 minimum to 100000 maximum)

Parameter: `reorg_batch_size`

This option determines the size of the transactions that the Client uses during a **Reorganize** command to set the value of newly-added columns to their initial values, as defined in the DASDL. The **Redefine** and **Customize** commands create a reorg script that uses a stored procedure to do the updates in batches that are executed as transactions. For large tables, this process can require more time, but it does not run the database out of log space. Consider using the internal clone option instead.

Do not set initial values for new columns

Parameter: `inhibit_init_values`

Inhibits the **Reorganize** command from setting new columns to their INITIALVALUE in DMSII. As a result, the new columns will start off as being NULL.

10.3.4 Customizing - History Tables

Use the following parameters to customize the selected data source before cloning.

Configuration file parameters are included below with the `following_font`.

Data set history tables

Parameter: `history_tables`

These buttons define the initial values of two data set property bits, which we show as History Tables (None, Save History, History Only). You can override a setting for individual data sets by changing the setting in the data set properties page.

- **None:** No history tables are created.
- **Save history:** Creates normal and history tables in the DATATABLES Client control table. The normal table replicates the host data set. The history table contains a record for every update, which includes the update type, the data for the update record, and one or more columns used to maintain the order in which the updates occurred. For the SQL Server Client, the IDENTITY column is ideally suited for this purpose. A data warehouse can receive the synchronized clone from the normal table first. History tables use the original table name with a "_h" appended to it. After periodically importing the history table into the data warehouse, the history table can be purged.
- **Save history only:** Creates only history tables. These tables receive cloned data and updates, in addition to records of the updates. The normal data table is not created or populated.

Options

Enable dynamic history

Parameter: `enable_dynamic_hist`

When enabled, history tables are added without recloning all of the affected data sets. You can specify the default history columns from the Customizing User Columns Section Two configuration page. Once the changes have been saved run a **Redefine with options** command from the **Advanced** button dropdown for the data source, and select the **Redefine all data sets (-R)** option. When the command completes, you'll need to run a **Reorganize** command to create the history tables and their indexes.

Inhibit Drop

Parameter: `inhibit_drop_history`

Prevents the Databridge Client from inadvertently dropping history tables during a `clone`, `process`, or `drop` command.

This is a safeguard to prevent an unrecoverable error. If you drop tables, it's recommended that you re-enable this option when you restart the Client.

10.3.5 Customizing - SQL Data Types

Use the following parameters to control the default data types used by the **Define/Redefine** and **Customize** commands.

Configuration file parameters are included below with the `following_font`.

Default SQL data types

Use bigint for integers greater than 32-bits

Parameter: `use_bigint`

(SQL Server only) This option, when enabled, indicates that the Databridge Client will map DMSII numeric items that are too large to fit in an **int** data type (32-bit integer), to columns whose data type is **bigint** (64-bit integer). If this option is disabled, columns will have data types of **decimal**(*n*) instead. Items that are too large to fit in a **bigint** are still mapped to **decimal**(*n*). This option is designed to avoid customizations when you want to standardize the tables to use the **bigint** data type instead of **decimal**(*n*) whenever possible.

Use date data type

Parameter: `use_date`

(SQL Server only) This option, when enabled, indicates that the Databridge Client will map dates that have no time part to columns whose data type is **date**. If this option is disabled, these columns will have a data type of **smalldatetime** instead. This eliminates the need to customize such items when you want to standardize the tables to use the **date** data type instead of **smalldattime**.

The **date** data type has a much bigger range than both the **smalldatatime** and **datetime** data types. It is the ideal data type for dates that have no time parts as it uses less storage than **smalldatetime**.

Use datetime2 data type

Parameter: `use_datetime2`

(SQL Server only) This option, when enabled, indicates that the Databridge Client will map date/time items to columns whose data type is **datetime2**. If this option is disabled, these columns will have a data type of **datetime** instead. This eliminates the need customize such items when you want to standardize the tables to use the **datetime2** data type instead of **datetime**.

The **datetime2** data type has a larger range than the **datetime** data type, and offers increased precision in the time part. The **datetime2** data type uses the same amount of storage as the **datetime** data type.

Use time data type

Parameter: `use_time`

(SQL Server only) This option, when enabled, indicates that the Databridge Client will map items that represent numeric times to columns whose data type is the **time** data type. If this option is disabled, such columns will have a data type of **int** instead. The value is formatted as *"hhmiss"*. This option is designed to avoid customizations when you want to standardize the tables to use the **time** data type.

> **Note**
>
> This option is not applicable to DMSII TIME(12) and TIME(14) which are stored in REAL in DMSII. They cannot be replicated to a column of data type **time**, as they can contain values that are bigger than 24 hours. These items are replicated to columns whose data types is **int**.

Use varchar, minimum length (0 to 255)

Parameter: `use_varchar` `min_varchar`

This option, when enabled, indicates that the Databridge Client will map character data to columns whose data type is **varchar** (Microsoft SQL Server) or **varchar2 (Oracle)** instead of **char**. When you enable **Use varchar**, you can also specify a minimum length (0 to 255).

> **Example**
>
> When this option is enabled, and a value of 4 is set for the minimum length input field, all DMSII ALPHA items whose length is less than 4 will be replicated to columns with the **char** data type. Using the **char** data type for narrow columns saves storage, as **varchar** data has a two byte length.

Use clob

Parameter: `use_clob`

This option indicates that DMSII ALPHA data that is larger than the 4000-character limit of the **varchar2** column should be mapped to a data type of **clob** instead of being truncated or split into two columns.

> **Note**
>
> Consider using the `Enable extended types` option, which is more efficient than this option.

Enable extended types

Parameter: `enable_extended_types`

(Oracle only) This option makes the Oracle Client read the Oracle database's `max_string_size` parameter and check if it is set to "extended" (as opposed to "standard").
If it finds the parameter set to "extended", it allows `varchar2` and `raw` columns to have a maximum length of 32K.

Use this parameter to ensure that DMSII ALPHA columns that are longer than 4000 characters do not get split. This option is more efficient than setting the parameter `use_clob` to True.

> **Caution**
>
> Before you can use this option you must alter the Oracle database and system to set the `max_string_size` parameter to extended. Consult the Oracle documentation for information on how to do this.
> Beware that once you set the database's `max_string_size` parameter to "extended", this step cannot be undone.

10.3.6 Customizing - SQL Suffixes

Use these options to create and assign suffixes to SQL statements that create indexes and tables. (These SQL statements are created by the **Generate Scripts** command.) By adding suffixes, you can add more SQL command specifications to the end of these SQL statements. You can reference these suffixes in the Relational Properties page of the **Customize** command.

Suffixes can be up to 256 characters in length. Configuration file parameters are included below with the `following_font`.

Table SQL Statement suffix

Default suffix (192 characters or less)

Parameter: `global_table_suffix`

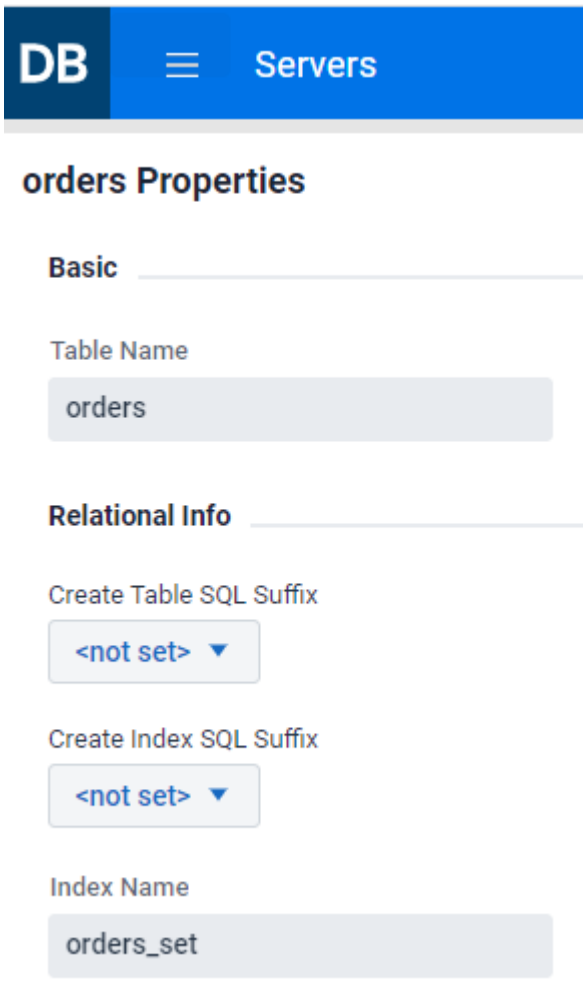
This option enables you to add a filegroup (SQL Server), tablespace (Oracle), or other SQL command specification to all of the `create table` SQL statements that the Client generates (except for statements associated with a specific suffix).

Index (1 to 100) Suffix

Parameter: `create_table_suffix` [*n*]

This option allows you to add a filegroup (SQL Server), tablespace (Oracle), or other SQL command specification for specific tables. This option sets the `create_suffix` column in the DATATABLES Client control table to the value *n*, which references the configuration `create_table_suffix` parameter with index *n*.

If using the **Customize** command, navigate to the Relational page for the data set and update the data table's properties by picking the desired suffix from the **Create table SQL Suffix** list box. For more information on user scripts, consult the *Databridge Client Administrator's Guide*.



Index SQL Statement suffix

Default suffix (192 characters or less)

Parameter: `global_index_suffix`

This option allows you to add a filegroup (SQL Server), tablespace (Oracle), or other SQL command specification to all of the `create index` SQL statements that the Client generates (except for statements associated with a specific suffix).

Index (1 to 100) Suffix

Parameter: `create_index_suffix` [*n*]

This option allows you to add a filegroup (SQL Server), tablespace (Oracle), or other SQL command specification for specific tables. This option sets the `index_suffix` column in the DATATABLES Client control table to the value *n*, which references the configuration `create_index_suffix` parameter with index *n*.

If using the **Customize** command, navigate to the Relational page for the data set and update the data table's properties by picking the desired suffix from the **Create index SQL Suffix** list box. For more information on user scripts, consult the *Databridge Client Administrator's Guide*.

Data masks

This section is only applicable to SQL Server Clients.

Index (1 to 100) Suffix

Parameter: `masking_parameter` [*n*]

(SQL Server only) Allows you to define the various parameter strings for the random and partial data masking functions.

If using the **Customize** command, navigate to the Relational page for the data set and update the desired data item to apply data masking on the data item properties page. After the masking function is selected, pick the desired masking parameter string from the **Masking Parameters** list box. For more information on user scripts, consult the *Databridge Client Administrator's Guide*.

10.3.7 Customizing - Translations

Use the following parameters to specify character translation settings.

Configuration file parameters are included below with the `following_font`.

> **Note**
>
> If you customize the character translation tables when you populate the data tables the first time, you must use them on all subsequent updates or the data will be invalid.

Use external translation library

Parameter: `eatran_dll_name`

Enables the translation of 16-bit character sets from EBCDIC to ASCII using an alternate data translation routine. The alternate data translation routine uses an external DLL, whose default name is `DBEATRAN.DLL` (`dbeatran.so` for UNIX) instead of the standard translation procedure. This DLL is dynamically loaded when this option is enabled. A Windows DLL named DBEATRAN.DLL is provided for translating Kanji data in DMSII to code page 932 for Japanese sites. If the DLL has a different name enter it in the provided edit box.

Configured Translations

Use this section to redefine how characters get translated. Enter the values that represent the EBCDIC character code and the corresponding ASCII character code in the edit boxes and select the **Add** button. The values will be included in the list generated below the input fields. All values must be entered as decimal numbers.

To delete one or more entries, select the corresponding check boxes in the list and select the **Delete** button.

You cannot specify characters that are constant across national characters. These characters include the space, hyphen (-), single quotation mark ('), digits (0-9), and the alphabet (A-Z and a-z).

10.3.8 Customizing - User Columns Section One

User columns let you add non-DMSII information to the relational database tables. For example, you can add an Audit Timestamp column to store the audit file timestamp and track when the data was last updated.

The options below allow you to change the default names, sql types and sql lengths (when applicable) for the various types of user columns. The external columns are split into two pages, the second page allows you to select the default user columns for the various types of tables.

The corresponding configuration options for entries in this page are `external_column` [*i*] = "*name*", *sql_type*, *sql_length*), where *i* is the index of the user column, *sql_type* is the index of the SQL type, and *sql_length* is the length of the column when the SQL type has a length.

User Column Default Properties

This table of user columns has 3 columns that you can modify,

Use the following options to modify the default specifications for user columns in your relational database tables.

Column Name

This column represents the default column name. Use the input fields in this column to provide a new name for the **Define/Redefine** and **Customize** commands to use when creating the columns in the DATAITEMS Client control table.

Data Type

This column allows you to select a data type for the column from the provided list of data types. Available data types are dependent on the user column type and the type of the relational database.

Data Length

Specify the length for the data type if a length is required. Most user columns impose a minimum and maximum length for the data type. If the value entered is outside the allowable range, you will get the error `"Invalid SQL Length"` and will then need to correct the length before the input will be accepted.

Available User Columns

The following is a list of the different types of user columns that are available. You can change the default name, data type, and data length using the controls on this page.

Audit Block Serial Number

A column that contains the Audit Block Serial Number (ABSN) of the block in the audit trail from which updates are currently being processed. If you use a decimal number its precision must be at least 10.

Audit File Number

A column that contains the Audit File Number (AFN). If you use a decimal number, its precision must be at least 4; otherwise, the value may be too large and result in a SQL error.

Audit Timestamp

A column that contains the audit file timestamp of the block from which updates are currently being processed. This value is stored using a date/time data type. For extract records this column is NULL.

Audit & Extract Timestamp

A column that contains the audit file timestamp of the block from which updates are currently being processed. This value is stored using a date/time data type. For extract records this column takes on the date/time value of when the data extraction started, instead of being NULL.

Create Time

The time when the record was created in the relational database (uses the data type **date** for Oracle and **datetime** or **datatime2** for SQL Server).

Data Source ID

A column that contains the data source identifier, as defined in the `data_source_id` column of the DATASOURCES client control table.

Data Source ID Key

This column is identical to Data Source ID except that it is used as a key.

Data Source Name

A column that contains the data source name.

Delete Sequence Number

This column augments the Deleted Record column with a sequence number to provide higher granularity and avoids the creation of duplicate deleted records.

Deleted Record

When this column is added to a table, deleted records are marked as deleted and left in the table. The client makes this column part of the index, which allows multiple instances of a deleted record to exist in the table and not be considered a duplicate. The value of this column is a timestamp that represents the number of seconds elapsed since the beginning of the epoch (1/1/1970).

Include the **Delete Sequence Number** column if this column does not provide enough granularity to avoid duplicate records (for example, a record is deleted twice in the same second).

Identity Column

(SQL Server only) Identifies the column using the sequence number assigned to the record when the record was created. Updates have no effect on this number.

Sequence Number

A sequence number used in history tables to determine the order of updates when they have the same update_time values. For SQL Server, the Identity Column for history tables is preferable, as it doesn't have this problem. For optimal results, let the client choose the default user columns for history tables.

Server Update Time

The time in which the update was applied to the relational database.

SQL Server Timestamp

Use this column (or the Identity column) for compatibility with older SQL Server databases. SQL timestamps are binary numbers that indicate the relative sequence in which data modifications took place in a database. The timestamp data type was originally implemented to support the SQL Server recovery algorithms.

Update Type

In history tables, this column indicates the type of update involved (insert, delete, or update). For non-history tables, this column indicates the type of the last update performed. (A value of 0 is assigned to this column during data extraction.)

Update Type (Logical Delete)

A column used in tables that preserves one deleted record per index value.

User Column 1

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

User Column 2

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

User Column 3

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

User Column 4

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

10.3.9 Customizing - User Columns Section Two

User columns let you add non-DMSII information to the relational database tables. For example, you can add an Audit Timestamp column to store the audit file timestamp to track when the data was last updated.

This page allows you to specify the default user columns for all primary tables, secondary tables, and history tables. This configuration page has a table with 4 columns, the first column contains the name of the user column, the next 3 columns labeled **Primary Tables**, **Secondary Tables** and **History Tables** represent the corresponding bits for these columns and the configuration paramters for entries in this page are `dflt_user_columns`, `sec_tab_column_mask` and `dflt_history_columns`.

The following is a list of the different types of user columns that are available. Options for each column (such as Add, Secondary Tables, and History Tables) are described in detail below.

User Column Options

Use the following options to include and modify user columns in your relational database tables:

- **Primary Tables**: Adds the selected user column to your relational database.

  If you change the name, SQL type, or SQL length using these settings, those changes will be in effect if you later add user columns in the data set properties using the **Customize** command.
- **Secondary Tables**: Includes the selected user column in any secondary tables.
- **History Tables**: Includes the selected user column in history tables.

Available User Columns

To add columns from the list below select the check boxes for the columns you wish to add to the corresponding types of tables (Primary Tables, Secondary Tables, and History Tables).

Audit Block Serial Number

A column that contains the Audit Block Serial Number (ABSN) of the block in the audit trail from which updates are currently being processed. If you use a decimal number its precision must be at least 10.

Audit File Number

A column that contains the Audit File Number (AFN). If you use a decimal number, its precision must be at least 4; otherwise, the value may be too large and result in a SQL error.

Audit Timestamp

A column that contains the audit file timestamp of the block from which updates are currently being processed. This value is stored using a date/time data type. For extract records this column as NULL.

> **Note**
>
> This column and the **Audit & Extract Timestamp** column are mutually exclusive, you will get an error if you try to include both options.

Audit & Extract Timestamp

A column that contains the audit file timestamp of the block from which updates are currently being processed. This value is stored using a date/time data type. For extract records, this column takes on the date/time value of when the data extraction started, instead of being NULL.

> **Note**
>
> This column and the **Audit Timestamp** column are mutually exclusive, you will get an error if you try to include both options.

Create Time

The time in which the record was created in the relational database (uses the data type **date** for Oracle and **datetime** or **datatime2** for SQL Server).

Data Source ID

A column that contains the data source identifier, as defined in the `data_source_id` column of the DATASOURCES client control table.

> **Note**
>
> This column and the **Data Source ID key** column are mutually exclusive, you will get an error if you try to include both options.

Data Source ID Key

This column is identical to Data Source ID except that it is used as a key.

> **Note**
>
> This column and the **Data Source ID** column are mutually exclusive, you will get an error if you try to include both options.

Data Source Name

A column that contains the data source name.

Delete Sequence Number

This column augments the Deleted Record column with a sequence number to provide higher granularity and avoids the creation of duplicate deleted records.

Deleted Record

When this column is added to a table, deleted records are marked as deleted and left in the table. The client makes this column part of the index, which allows multiple instances of a deleted record to exist in the table and not be considered a duplicate. The value of this column is a timestamp that represents the number of seconds elapsed since the beginning of the epoch (1/1/1970).

Include the **Delete Sequence Number** column if this column does not provide enough granularity to avoid duplicate records (for example, a record is deleted twice in the same second).

Identity Column

(SQL Server only) Identifies the column using the sequence number assigned to the record when the record was created. Updates have no effect on this number.

Sequence Number

A sequence number used in history tables to determine the order of updates when they have the same update_time values. For SQL Server, the Identity Column for history tables is preferable, as it doesn't have this problem. For optimal results, let the client choose the default user columns for history tables.

Server Update Time

Time in which the update was applied to the relational database.

SQL Server Timestamp

Use this column (or the Identity column) for compatibility with older SQL Server databases. SQL timestamps are binary numbers that indicate the relative sequence in which data modifications

took place in a database. The timestamp data type was originally implemented to support the SQL Server recovery algorithms.

Update Type

In history tables, this column indicates the type of update involved (insert, delete, or update). For non-history tables, this column indicates the type of the last update performed. (A value of 0 is assigned to this column during data extraction.)

> **Note**
>
> This column and the **Update Type (Logical Delete)** column are mutually exclusive, you will get an error if you try to include both options.

Update Type (Logical Delete)

A column used in tables that preserve one deleted record per index value.

> **Note**
>
> This column and the **Update Type** column are mutually exclusive, you will get an error if you try to include both options.

User Column 1

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

User Column 2

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

User Column 3

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

User Column 4

A generic user column whose entry is NULL. To add a default value to this type of column, use the Data table creation user script. For more information, see the *Databridge Client Administrator's Guide*.

10.3.10 Customizing - User Scripts

Use the following options to specify locations where you can save and archive user scripts.

Parameters are included below with the `following_font`.

Check for missing user scripts

Parameter: `check_user_scripts`

This option instructs the Databridge Client to return a warning if the "create table" and "create index" user scripts cannot be found and executed. User scripts are executed after the Databridge Client scripts are used to create tables and indexes during **Process** and **Clone** commands.

User scripts directory (192 characters or less)

Parameter: `user_script_dir`

Specifies the location where data source customizations are saved as user scripts. If a problem occurs, you can use these scripts to restore your Client control table customizations using a **Define/Redefine** command. This parameter defaults to "scripts", which is a sub-directory of the Client's working directory.

When file security is enabled, the user scripts directory must be inside the Databridge Client's working directory to prevent unauthorized access. If you have common scripts among the various data sources in the working directory you could set the User scripts directory to `..\UserScripts`. If this is done, you must manually create the directory.

User scripts archive directory (192 characters or less)

Parameter: `user_scr_backup`

(Optional) Specifies the location to which archival copies of user scripts are saved when you run the `Create Scripts` command. If a directory is not specified, scripts are archived to subdirectories of the user script directory.

10.4 Logging

10.4.1 Logging

You can configure settings for

- Client log files
- Service log files
- Trace log files

10.4.2 Logging/Tracing - Client Log

Use this page to configure the settings for the client log files. The Client programs create log files in the "logs" subdirectory of the working directory for the corresponding data source. Both client programs (**DBClient** and **dbutility**) use the main log file, which is named *prefix* `YYYYMMDD[_HHMMSS].log`. The time part is only included in the name when a log switch occurs and the new and old file names are the same.

The **DBCIntCfgServer** program uses a different log file that has "_cfg" appended to the *prefix*. The **makefilter** utility also uses a different log file that has "_flt" appended to the *prefix*.

Log file parameters are included below with the `following_font`.

Generate single line output messages in log file

Parameter: `single_line_log_msgs`

This option causes the client to generate all log output as single-line messages. This option exists to assist some log file analysis programs that fail to parse multi-line output messages.

Start new log on new day

Parameter: `newfile_on_newday`

This option makes the Client create a new log file upon start up if the last log file was created on an earlier date.

Switch log daily

Parameter: `logsw_on_newday`

This option makes the Client switch to using a new log file if the date changes while it is running.

Switch log on size

Parameter: `logsw_on_size`

This option makes the Client switch to using a new log file if the current log file reaches the specified size limit. If new and old log files have the same creation date, a time stamp is included in the filename to ensure that it is identifiable.

Maximum file size

Parameter: `max_file_size`

Specifies the size limit of Client log files. Enter a number with a suffix of K, M and G to indicate the unit of measure (kilobyte, megabyte, or gigabyte). When enabled, an input field will be available to enter the desired size limit.

File name prefix (20 characters or less)

Parameter: `file_name_prefix`

This option specifies a string (up to 20 characters in length) that replaces the default prefix "db" for future log files. This can help you manage your log files if you have multiple data sources. The specified prefix is applied to both Client log files and **DBCIntCfgServer** log files. It is recommended to use the data source name as the prefix for the log files, as it makes managing log files when working with multiple data sources easier.

10.4.3 Logging/Tracing - Service Log

Use this page to configure settings for service log files. Service log files store all activity related to the service and are created in the logs subdirectory of the service's working directory. The log files are named *prefix* `YYYYMMDD*[_*HHMMSS*].log`. The time part is only included in the name when a log switch occurs and the new and old file names are the same.

Log file parameters are included below with the `following_font`.

Start new log on new day

Parameter: `newfile_on_newday`

This option makes the service create a new log file upon start up if the last log file was created on an earlier date.

Switch log daily

Parameter: `logsw_on_newday`

This option makes the service switch to using a new log file if the date changes while the service is running.

Switch log on size

Parameter: `logsw_on_size`

This option makes the service switch to using a new log file if the current log file reaches the specified size limit. If the new and old log files have the same creation date, a time stamp is included in the filename to ensure that it is identifiable.

Filename prefix

Parameter: `file_name_prefix`

This option specifies a string (up to 20 characters in length) that replaces the default prefix "cp" for future log files.

Maximum file size

Parameter: `max_file_size`

Specifies the size limit of Service log files. Enter a number with a suffix of K, M and G to indicate the unit of measure (kilobyte, megabyte, or gigabyte). When enabled, an input field will be available to enter the desired size limit.

10.4.4 Logging/Tracing - Trace

Use this page to configure settings for trace files. To specify activities to trace, use the Default Trace and Log Options dialog box by select the **Advanced** button on the data sources page and selecting **Trace and Log Options**.

Both Client programs (**DBClient** and **dbutility**) use a trace file (*prefix* `YYYYMMDD*[_*HHMMSS*].log`). The time part is only included in the name when a trace file switch occurs and the new and old file names are the same. The **DBClntCfgServer** program uses a different log file with "_cfg" appended to the *prefix*. The **makefilter** utility, also uses a different log file with "_flt" appended to the *prefix*.

Trace file parameters are included below with the `following_font`.

Maximum file size**

Parameter: `max_file_size`

Use this option to specify the size limit of trace files. Enter a number with a suffix of K, M and G to indicate the unit of measure (kilobyte, megabyte, or gigabyte). When enabled, an input field will be available to enter the desired size limit. When the configured size limit is reached, a new trace file is generated with a time part prefix in the file name.

Filename prefix

Parameter: `file_name_prefix`

Use this option to specify a string (up to 20 characters in length) to replace the default prefix "trace" for future trace files.

10.5 Processing

10.5.1 Processing

The following configurations are available.

- General
- Advanced
- Date and Time
- Engine and Enterprise Server
- DMSII Data Error Handling
- Error Recovery
- Schedule
- Statistics
- Stop Conditions

10.5.2 Processing - General

This page defines run time options that do not require running a **Configure** or **Redefine** command to make them take effect.

Configuration file parameters are included below with the `following_font`.

Display only active data sets in display command

Parameter: `display_active_only`

This option affects the **Log Control Tables** command in the data source's **Advanced** button dropdown (equivalent to the **display** command when using the command-line Client).

When this option is enabled the command only shows the Client control table entries for data sets whose `active` column is 1. This results in a lot less output when you have a lot of data sets whose `active` column is 0. If this option is disabled, the control table entries for all data sets are displayed.

Number of auxiliary (ODBC/OCI) statements (0 minimum to 200 maximum)

Parameter: `aux_stmts`

Specifies the number of database API (that is, ODBC or OCI) statements that can be assigned to individual SQL statements. Using multiple statements allows SQL statements to be parsed once and executed multiple times, provided that the statement is not reassigned to hold another SQL statement. Increasing the number of database API statements significantly reduces processing time.

Audit unavailable action

Wait and retry

Parameter: `use_dbwait`

When this option is enabled, the Databridge Engine (DBEngine or Enterprise) enters a wait-and-retry loop when it reaches the end of the audit trail, instead of stopping the Client with an end-of-audit status. Enable this option if you use real/time replication.

Use the following two parameters to control the wait-and-retry loop:

- **Retry interval (in seconds)**

  Parameter: `max_retry_secs`

  Specifies the time interval between retries by the Databridge Engine when it is in a wait-and-retry loop. If the Databridge Engine finds new entries in the audit, it will start processing from where it left off. Otherwise, it will wait until the next retry time and try again.

- **Maximum wait time (in seconds)**

  Parameter: `max_wait_secs`

  Specifies the maximum amount of time that the Databridge Engine stays in a wait-and-retry loop before it stops the Client. A value of 0 for this parameter is interpreted as a request to the Databridge Engine to wait and retry forever.

- **Incremental wait time (in seconds)**

  Parameter: `max_wait_secs`

  Enabling this option makes an input field visible. This input field is used to specify the incremental value of **max_wait_secs** that is optionally used to break up large wait times into smaller increments by making the Client repeatedly issue DBWAIT calls using this second value. The second value, which must be smaller than the first value (unless the first value is 0). For example, setting **max_wait_secs** to 3600,60 will result in the Client issuing a DBWAIT remote procedure call with a `max_wait_secs` value of 60 seconds.

  Upon getting a `"no more audit available"` return status, the Client will issue another DBWAIT call until it has received no updates for the amount of time indicated by the first parameter. This method ensures that an idle line has traffic, which makes it possible to detect situations where the network goes down and neither side knows about it. Upon receiving an update the Client resets the timer that keeps track of the time during which no updates are received. A value of 0 for this option makes the Databridge Engine handle the wait-and-retry loop without any involvement by Client.

Store NULL DMSII numbers as

Paramter: `null_digit_value`

000... Stores NULL DMSII numbers as zeros (0).

999... Stores NULL DMSII numbers as nines (9).

> ⚠ **Caution**
>
> Do not change this parameter after you have cloned the DMSII database, as numeric columns that do not allow nulls will get different values when the data is NULL.

10.5.3 Processing - Advanced

This page defines and allows the configuration of advanced run time options. Most of which do not require the running a **Configure** or **Redefine** command to make them take effect.

Configuration file parameters are included below with the `following_font`.

General

Automated virtual data sets

Parameter: `automate_virtuals`

> **Note**
>
> This option must be enabled before you clone the database.

This option enables code that automatically handles virtual data sets that are linked with their parent data sets using the `virtual_ds_num`, `real_ds_num`, and `real_ds_rectype` columns in the DATASETS Client control table. When using the **Customize** command, select the properties icon  for the selected dataset to set these links in the **Virtual Dataset** group. Alternatively, this can be done using user scripts.

These links allow **Clone** commands to not require that virtual data sets be explicitly specified on the command line. For example, if the virtual data set LN-HISTORY-REMAP is linked to the data set LOAN. Re-cloning LOAN will also clone LN-HISTORY-REMAP, even if it is not specified on the command line.

Override changed columns only option

Parameter: `enable_minimized_col`

When the **Update changed columns only** option is enabled and the **Customize** or **Define/Redefine** command is run, the corresponding bit is set in the `ds_options` column of the DATASETS table for all the data sets whose `active` column is 1. This option, when disabled, allows you to make the Client ignore this bit, which effectively disables the **Update changed columns only** option. Enabling this option will cancel the override. This option was implemented so running a **Redefine** command would not be required if the **Update changed columns only** option has caused a problem.

Override optimized SQL updates option

Parameter: `enable_optimized_sql`

When the **Optimize SQL updates** option is enabled and the **Customize** or the **Define/Redefine** command is run, the DSOPT_Use_bi_ai bit is set in the `ds_options` column of the DATASETS Client

control table for all the data sets whose `active` column is 1. This option, when disabled allows you to make the Client ignore the DSOPT_Optimize_4_CDC bit. Enabling this option will cancel the override. This option was implemented so running a **Redefine** command would not be required.

Preserve deleted records during a reclone

Parameter: `preserve_deletes`

This option, when enabled allows the Client to preserve deleted records when a data set that preserves deleted records is re-cloned. Instead of dropping and recreating the table, the Client runs a cleanup script that removes all records, except for deleted records.

> **Caution**
>
> If the layout of the tables has changed this option will not be available. Allow the Client to reorganize the table before re-cloning it.

Multithreaded updates

Number of DMSII buffers (2 minimum to 64 maximum)

Parameter: `n_dmsii_buffers`

This option specifies the number of buffers to be used by the Client in communications with the server. Raising this value may improve performance by ensuring there are enough buffers queued to keep the update workers busy at all times. Letting this option default causes the Client to determine the value needed. It is recommended to use 2 DMSII buffers when running in single-thread update mode. In multi-threaded update mode, it is recommended to use more than 2 buffers per update thread. The recommended values are 8-threads and 64-buffers.

Number of update threads (2 minimum to 16 maximum)

Parameter: `n_update_threads`

This option specifies the number of update threads, which are responsible for executing SQL to update the user tables and writing bulk loader temporary files. When using the BCP API in the SQL Server Client, these threads are also responsible for making the BCP API calls to load the data. If you have a powerful machine, setting this parameter to a high value will increase the update processing speed at the expense of additional memory. Avoid setting this parameter to 1, as this effectively passes off all updates to the single worker thread, when executing updates directly is recommended.

SQL execution timeout values (in seconds)

Query alert threshold

Parameter: `sql_exec_timeout` - first value

This option allows you to override the default setting of 3 minutes (180 seconds), for the time after which the Client issues a WARNING about the query taking too long to complete.

Query abort threshold

Parameter: `sql_exec_timeout` - second value

Enabling this option will make an input field visible. This input field is used to specify a secondary timeout value for a SQL query after which time the query is aborted. This value must be greater than the value specified for the **Query alert threshold** described above.

Generate SQL heartbeats (in minutes; 15 minimum to 300 maximum)

Parameter: `sql_heart_beat`

This option was implemented as a work-around for the situation where long clones result in the Client's connection to the database getting closed, as a result of long periods of inactivity. When this option is enabled, the client periodically executes a dummy SQL update on the Client's connections to the database to keep the connections alive during the data extraction where the only activity is on the bulk loader connection.

Enabling this option will make an input field visible. This input field is used to specify the time interval between SQL heartbeats.

Server inactivity timeout (in minutes)

Parameter: `max_srv_idle_time`

Enabling this option will make an input field visible. This input-field is used to specify a timeout value for aborting the server connection after several inactivity WARNINGS. You should use a value large enough to see one of the WARNINGS. The Client issues a server inactivity warnings after 5 minutes of inactivity, the second warning is after 15 minutes of inactivity, and the third one is after 30 minutes of inactivity (the WARNING is repeated every 30 minutes from this point)

Note

There are two type of server inactivity WARNINGs:
`"No data received from server for *nn* minutes"` and
`"No data received from server for *nn* minutes"`. This option is associated with the first warning, which indicates that the server is not responding to the RPC issued by the Client. The second warning indicates that the Client has not received any updates from the server, which could be an indication that the database is idle.

10.5.4 Processing - Date and Time

This configuration page defines options for items that are date and time values in DMSII.

Configuration file parameters are included below with the `following_font`.

Date parameters

Century break dynamic when off (yy)(00 to 99)

Parameter: `century_break`

Several DMSII date formats have 2-digit years (for example "mmddyy"). This option determines the algorithm for changing the value of "yy" to a 4-digit year. For example, if **Century break** is set to 50, values less than 50 are 21st century years (20yy) and values greater than or equal to 50 are 20th century years (19yy). This method does not work very well as the century break needs to be adjusted from time to time so the current year is in the middle of the range. The Client supports a dynamic century break, where the client uses the current year to set the century break.

When this option is disabled, the century break is dynamically computed when the Client starts. To set it to a different value you need to enable this option, which makes the input field below the option available. Enter the desired value in this input field and select **Save** at the bottom right of the page.

LINC date base year (yyyy)(1800 to 2100)

Parameter: `linc_century_base`

Specifies the base year for LINC database dates as a four-digit year value.

> **Caution**
>
> Do not change this parameter after you have cloned the DMSII database, as you will get different results when decoding Linc dates.

Set LINC date 0 to NULL

Parameter: `set_lincday0_to_null`

When enabled, this option indicates that a LINC date value of 0 should be treated as NULL, rather than 1/1 of the LINC date base year.

Correct invalid date values

Parameter: `correct_bad_days`

This group of buttons defines the method in which bad dates are corrected. The four options available are mutually exclusive.

- **Do not correct:** the Client does correct any bad dates.
- **Only correct days that are 0:** the Client corrects dates with a day value of 0 by changing the day value to 1. This is useful if you have dates that do not have a day part (like credit card expiration dates).
- **Correct only bad days <= 31:** the Client attempts to correct DMSII dates with a bad day value by setting the value to last day for the given month and year (unless the day value is greater than 31).
- **Correct all bad days and months:** the Client performs the following corrections in addition to the ones handled by **Correct only bad days <= 31**. Day values greater than 31 are set to the last legal day of the month, day values that are 0 are set to 1, month values greater than 12 are set to 12, and a month value of 0 is set to 1.

Default formats

Numeric date

Parameter: `numeric_date_format`

Specifies the date format used to encode numeric dates. Options include all six and eight-digit numeric date formats.

Default date format

Parameter: `default_date_fmt`

Specifies the default date format that appears when you select the **Replicate as Date** option in the DMS item properties page when using the **Customize** command.

Null date values for columns that do not allow NULL

The options in this section only apply to the SQL Server Client.

Datetime

Allows you to change the value used to represent a NULL date in a **datetime** column that does not allow nulls.

Datetime2 and date

Allows you to change the value used to represent a NULL date in a **datetime2** column that does not allow nulls.

10.5.5 Processing - DBEngine and DBEnterprise

Most of the options in this section allow the Client to override the configured values of the corresponding parameters in the Engine control file.

Configuration file parameters are included below with the `following_font`.

General

DBEngine (Extract) workers (1 minimum to 10 maximum)

Parameter: `engine_workers`

This option overrides the DBEngine WORKERS=*n* parameter on the host, which controls the number of extract workers the Databridge Engine can use during the data extraction phase.

Convert reversals to updates

Parameter: `convert_reversals`

The buttons **Use server setting**, **Don't convert**, and **Convert** allow the Client to override the Databridge Engine control file parameter CONVERT REVERSALS. Refer to the *Databridge Host Administrator's Guide* for more information on this parameter.

Enable DOC records (for debugging only)

Parameter: `enable_doc_records`

This option, when enabled, requests DOC records from the Databridge Engine (DBEngine or Enterprise Server). Use this option only for troubleshooting Databridge Engine problems.

Include latest StateInfo in record headers

Parameter: `use_latest_si`

This option, when enabled, requests StateInfo in all data records sent by the server during audit file processing. If the DMSII audit file records have an associated timestamp, this option can result in more accurate audit timestamp values.

Enterprise Server audit file origin

Parameter: `dbe_dflt_origin`

This option, when enabled, specifies the expected origin for Enterprise Server audit files during normal operations. Options include: **Cached**, **Direct disk**, and **Indirect disk**.

COMMIT parameters

Use these options to override Databridge Engine CHECKPOINT FREQUENCY parameters.

COMMIT PARAMETERS

Every (n) BLOCKS (0 minimum to 200000 maximum)

Parameter: `commit_absn_inc` (0 minimum to 200000 maximum)

Specify a value to override the Databridge Engine CHECKPOINT CLIENT EVERY *nnn* AUDIT BLOCKS parameter setting. The Databridge Engine will generate a commit at the next quiet point after the specified number of audit blocks have been processed since the last commit.

Every (n) UPDATES (0 minimum to 200000 maximum)

Paramter: `commit_update_inc`

Specify a value to override the Databridge Engine CHECKPOINT CLIENT EVERY *nnn* UPDATE RECORDS parameter setting. The Databridge Engine will generate a commit at the next quiet point after the specified number of updates have been processed since the last commit.

Every (n) SECONDS (0 minimum to 300 maximum)

Parameter: `commit_time_inc`

Specify a value to override the Databridge Engine CHECKPOINT EVERY *nnn* SECONDS parameter setting. The Databridge Engine will generate a commit at the next quiet point after the audit timestamp has moved by specified number of seconds since the last commit.

Commit during long transactions

Paramter: `commit_longtrans`

To have the Client override the Databridge Engine CHECKPOINT LONG TRANSACTIONS parameter at the start of a process command, select either **Enable** (to set the parameter to True) or **Disable** (to set the parameter to False). If you don't want the Client to override the parameter, select **Use server setting**.

Commit during idle database

Paramter: `commit_idle_database`

This option allows the Databridge Client to override the Databridge Engine COMMIT DURING IDLE DATABASE parameter setting in the Engine's control file. Select either **Enable** (to set the parameter to True) or **Disable** (to set the parameter to False). If you don't want the Client to override the parameter, select **Use server setting**.

BATCH COMMIT PARAMETERS

Batch job (start time, end time)

Paramter: `batch_job_period`

This option specifies the block of time during which batch jobs typically run. For example "22:00, 01:00" indicates that batch jobs run between 10:00 pm and 1:00 am. The syntax for the four commit checkpoint parameters, that can be overridden by the client, was modified to allow an optional second value to be specified. The second value represents the alternate value to be used during the batch period. The client was then modified to implement the automatic switching of commit parameters between the two periods. The switching is based on the value of the audit time stamp rather than the time when the client is run.

When you enable the **Batch job (start time, end time)** option two time input fields and a group labeled **Commit frequency** become visible below the enabled option. Use the input fields to set the batch period times and set the commit frequency parameter value for the batch period using the control in the **Commit frequency** group. You do not need to set all four Commit Frequency options as the disabled options will honor the value in the Engine control file.

10.5.6 Processing - DMSII Data Error Handling

This page contains parameter related to handling in errors in DMSII data.

Configuration file parameters are included below with the `following_font`.

Character data error

Control character

The Client can be configured to handle control characters in ALPHA items in one of the following ways that are mutually exclusive. The following buttons allow you to select how you want to handle control characters.

- **Change to space**

  Parameter: `convert_ctrl_char`

  The Client will change all control characters encountered in ALPHA data to spaces. This option will require that you run a **Configure** or a **Redefine** command to update the `da_options` column in the DATAITEMS Client control table entries that have a character data type. If you use a **Redefine** command you need to use the **Redefine with options** command from the **Advanced** button dropdown for the data source and select the **Redefine all data sets (-R)** option.

- **Change to '?' and count as error Parameter:** `inhibit_ctrl_chars`

  The Client will treat a control character encountered in alpha data as an error and convert it to a question mark (?). If the percentage of bad character in the item exceeds the configured threshold, the item will be set to NULL.

- **Translate if possible**

  The Client will translate control characters that do not interfere with Client operations. The Client won't allow a control character to be translated to the character NULL (00) as this causes problems with NULL terminated strings. In addition, it won't allow the use of CR and LF characters, as they interfere with bulk-loader operations. If you use SQL Server and the TAB character is the bcp delimiter, the Client won't allow you to translate a character to a TAB character. These characters will be changed to question marks (?) and counted as errors.

Change 8-bit character to '?' and count as error

Parameter: `inhibit_8_bit_data`

Enabling this option makes the Client treat 8-bit characters in ALPHA data as an error and converts them question marks (?).

Keep undigits in NUMBER items stored as character data

Parameter: `keep_undigits`

An undigit is a digit in a DMSII NUMBER item, whose value is not in the range 0 to 9.

Setting this option to 1 or 2 causes the Client to preserve the bad digit values in DMSII NUMBER items that are stored as CHAR or VARCHAR data. The bad digits will have values of A-F.
Setting it to 2 also causes undigits in DMSII NUMBER items to be treated as 9s. If the option is set to 0, undigits in DMSII NUMBER items are treated as errors.

Enable High Value Padding

Parameter: `enable_ff_padding`

This option lets you mark items as padded with high values in order to achieve left justification. It applies to ALPHA items and unsigned numeric items that are stored as ALPHA data.

Set item to NULL if errors exceed (%)

Parameter: `alpha_error_cutoff`

Specify the percentage of data errors in any given alpha item that is tolerated before the item is declared bad and treated as NULL.

General error handling

Discard records containing data errors

Parameter: `discard_data_errors`

Enable this option to instruct the Client to write all records with data errors to the discard file `tablename.bad`, and not apply them to the relational database.

Display data errors (in log file)

Parameter: `display_bad_data`

When this option is enabled, the Databridge Client will display the raw DMSII data for an item that has a data error. Use this option for debugging purposes if you encounter many data errors. This output is suppressed when the number of errors exceeds the limit specified by **Error display limit** values.

Suppress duplicate (insert) warnings

Parameter: `suppress_dup_warnings`

Enabling this option prevents warnings for duplicate inserts and failed updates from displaying during update processing. Enabling this option is not recommended as it will tend to mask the errors, which could be a symptom of a more serious problem.

Error display limits

Parameter: `error_display_limits`

Specifies the maximum number of errors per data set that are permitted for screen output (Display) and for log file output (Log file).

DISCARD RECORD THRESHOLDS

These values control how the Client handles discarded records.

Maximum total discards (0 minimum to 10000 maximum)

Specifies the total number of discards the Client will tolerate before abending. This value must be greater than the value for **Maximum per-table discards** unless it set to 0, which means that no limits are imposed on discards.

Maximum per-table discards (0 minimum to 1000 maximum)

Specify the maximum number of discards records for a table that is written to the discard file. Discards that exceed this number are ignored. If this option is set to zero, no limits are imposed on the discards for the individual table.

10.5.7 Processing - Error Recovery

This page contains the values for error recovery parameters for the data source in the Client Manager service's configuration file.

Configuration file parameters are included below with the `following_font`.

Options

Maximum retries

Parameter: `max_retries`

Specifies the number of times the service tries to run a `process` command that terminates with a recoverable error exit status. If the exit status indicates that the Databridge Server is inaccessible or that the relational database is down, this parameter is ignored and the service retries with longer intervals until the interval reaches five minutes, after which it retries every 5 minutes.

Minimum wait (in seconds)

Parameter: `sched_min_wait`

Specifies the minimum delay (in seconds) before a `process` command is attempted after a run terminates. It is designed to inject a delay between runs when a `process` command run terminates a few seconds before the next scheduled `process` command's launch time. This parameter is not shown in the Client Manager service's export file when its value is 0.

Retry interval (in seconds)

Parameter: `sched_retry_secs`

Specifies the interval (in seconds) between attempts by the service to launch the next `process` command when a `process` command terminates with a recoverable exit code.

Disable data source conditions

Parameter: `disable_on_exitcode`

Specifies a set of DBClient exit statuses that cause the service to disable the data source. For example, if you use STOP AFTER TASK *xxx* to detect the end of the banking day, you can make the Client disable the data source to prevent any further client activity when this run is detected in the audit trail. This allows you to run a program to create the end of day banking reports, while the relational database is frozen. When the report software finishes, you can enable the data source and resume change tracking.

The three conditions in question are **Stop task name reached**, which corresponds to the example above, **Stop time reached** which happens when you use STOP AFTER TIME *xxx*, and **Audit file**

number reached, which happens when you set the Client to stop after a given audit file is processed.

10.5.8 Processing - Schedule

This page contains the values for scheduling parameters for the data source in the Client Manager service's configuration file.

Configuration file parameters are included below with the `following_font`.

Schedule

Scheduling options apply only to `process` commands that are initiated by the service. The three buttons described below, allow you to schedule `process` commands runs at specific times of the day or a fixed amount of time after a `process` command run terminates. Selecting one of the last two scheduling options enables the further controls specific to the enabled option(s).

- **None**

  Indicates that the service will not use any scheduling. The Client runs are manually launched from the Administrative Console.

- **Fixed delay (in seconds)**

  Parameter: `sched_delay_secs`

  Indicates that this data source will use fixed delay scheduling, The input field specifies the delay (in seconds) between the time one Client run stops and another one begins for the same data source.

- **Daily (12 maximum)**

  Parameter: `daily`

  Using the time picker control provided, select or type the time at which the service should start a `process` command run. Then select the **Add** button. The time entered will appear in the **Scheduled Times** group. You can specify up to 12 times per day for scheduled `process` runs. The times provided do not to be in chronological order, as they will be sorted when added to the list.

  If a `process` command is already running at these times, the service will ignore the scheduling request.

Blackout period

Parameter: `blackout_period`

When enabled, two input fields will be available to specify times. Using the two input fields, specify the start and end times of a fixed block of time during which the Client cannot run. This option is useful for operations, such as database backups, that can only take place when the Client is inactive. For example, if you want to back up the database daily between 1:00 a.m. and 2:30 a.m. daily, define a blackout period from 0:55 to 2:30. The extra 5 minutes ensures that the Client finishes any long transactions before the database backup begins.

Automatic Run Options

Process data source on service startup

Parameter: `run_at_startup`

Enable this option so the service will start a `process` command run when the service is started.

Redefine data source when DMSII reorganization detected

Parameter: `auto_redefine`

Enable this option to continue processing without any interruptions after a DMSII reorganization, provided the reorganization did not affect the table layouts. When the exit code for a `process` command indicates that a DMSII reorganization has been encountered, the service starts **DBCIntCfgServer** run that executes a `redefine` command to update the Client control tables. If the `redefine` command determines that the DMSII reorganization did not affect the table layouts, the `redefine` command returns an exit status of 0, and the service immediately launches a `process` command.

If the `redefine` command determines that new scripts need to be generated, manual intervention is required. To have the Client automatically generate scripts in these cases, select **Generate scripts after an automatic redefine** in addition to this option.

There is no option to automate the running of a `reorganize` command, as it is not a good idea to run this command without first looking at what it will do, as altering big tables can take a very long time and possibly fill the database log, which can trigger a massive rollback.

Generate scripts after an automatic redefine

Parameter: `auto_generate`

Enable this option to have **DBCIntCfgServer** run a `generate` command when the exit code for a `redefine` or `process` command specifies that a `generate` command is required. This command creates a new set of scripts for the tables of data sets that require them. This usually indicates that the data set has been added or that the reorganization caused the data set to be re-cloned.

> **Note**
>
> If the exit status for the `redefine` command indicates that a `generate` command is required, this means that one or more data sets will be re-cloned. If you do not want clone commands automatically started, refrain from setting this option.

10.5.9 Processing - Statistics

This page allows you to configure run time parameters that control the statistics that get written to the log file during `process` and `clone` commands.

Configuration file parameters are included below with the `following_font`.

Logging options

Show performance statistics

Parameter: `show_perf_stats`

When enabled, this option enables the logging of performance statistics at the end of the data extraction phase, after the processing of an audit file finishes (i.e. when audit file switch occurs) and at the end of the run.

Show statistics

Parameter: `show_stats`

When enabled, this option indicates that record count statistics will be displayed at intervals that you specify using the parameter **Record count display intervals**. This can be useful for monitoring the progress of lengthy operations.

Show table statistics

Parameter: `show_table_stats`

When enabled, this option indicates that table statistics will be displayed at the end of the data extraction phase and after an audit file switch. During update processing these statistics also include the average update times.

Audit file statistics

Write statistics into the relational database

Parameter: `enable_af_stats`

Enabling this option causes a `process` command to save the end of audit statistics in the AF_STATS Client control table, which was added in Databridge 7.0. This table can hold up to 9999 records, which corresponds to audit files 1 through 9999.

The AF_STATS tables contains the following columns

Column Name	SQL Server data type	Oracle data type
data_source	char(30)	char(30)
audit_filenum	int	number(10)
no_stats_available	bit	number(1)
audit_start_time	datetime	date
client_start_time	datetime	date
client_end_time	datetime	date
n_threads,	int	number(10)
elapsed	int	number(10)
dms_rec_count	int	number(10)
sql_op_count	int	number(10)
sql_rb_op_count	int	number(10)
sql_suppressed	int	number(10)
sql_filtered	int	number(10)
recs_discarded	int	number(10)
recs_in_error	int	number(10)
bytes_received	bigint	number(15)
total_bytes_received	bigint	number(15)
bi_bytes_received	bigint	number(15)
create_count	int	number(10)
delete_count	int	number(10)
modify_count	int	number(10)
modify_bi_count	int	number(10)
modify_ai_count	int	number(10)
link_ai_count	int	number(10)
state_count	int	number(10)
doc_count	int	number(10)

Column Name	SQL Server data type	Oracle data type
commit_count	int	number(10)
rollback_count	int	number(10)

For a description of these columns see the *Databridge Client Administrator's Guide*.

Record count display intervals

Data extraction (1 minimum to 10000000)

Parameter: `statistics_increment`

Specifies the frequency at which records counts are displayed during data extractions.

Update processing (1 minimum to 100000)

Parameter: `statistics_increment`

Specifies the frequency at which records counts are displayed during update processing.

10.5.10 Processing - Stop Conditions

This page allows you to configure the various stop conditions for **Process** and **Clone** commands.

Configuration file parameters are included below with the `following_font`.

Stop after extraction phase (defer fixups)

Parameter: `defer_fixup_phase`

Enable this option to defer the fixup phase to the next **Process** command. The Client stops at the end of the data extraction phase.

Stop after fixup phase

Parameter: `stop_after_fixups`

When enabled, this option stops the Client when the fixup phase is complete. At that point, all data sets will have a `ds_mode` value of 2 and all of the tables in the relational database will be consistent.

Stop after garbage collection reorganization

Parameter: `stop_after_gc_reorg`

Enable this option to stop the Client at the first quiet point after a garbage collection reorganization.

Stop on Enterprise Server audit file origin change

Parameter: `stop_on_dbe_mode_chg`

Enabling this option causes the Client to stop as soon as it detects that the **Databridge Enterprise Server access mode** changed from the value specified in the parameter **Verify DBEnterprise audit origin**. If this parameter is set to **direct** and Enterprise Server switches to **indirect**, this will result in the Client stopping at the next quiet point.

Dynamic stop conditions

Stop task name

Parameter: `stop_condition_task`

Causes the Databridge Engine to stop processing when it encounters the specified task in the audit trail. This task must be a DMSII program (that is, a program that opens the DMSII database). The **After** or **Before** options indicate whether processing stops at the first quiet point *after* or *before* the task is encountered in the audit trail. Consult with DMSII personnel to determine the syntax of the task name.

Stop time

Parameter: `stop_condition_time`

Causes the Databridge Engine to stop processing when it encounters an audit block whose timestamp value is greater than or equal to the specified time. The Client automatically provides the date part of the time stamp based on the current audit location being processed. The **After** or **Before** options indicate whether processing should stop at the first quiet point *after* or *before* the audit block whose timestamp value causes the Databridge Engine to stop processing.

10.6 PCSpan

10.6.1 PCSpan

See the options to configure the Flat File client.

- General
- Advanced

10.6.2 PCSpan - General

Use the following options to customize the parameters specific to a Flat file data source.

Configuration file parameters are included below with the `following_font`.

Options

Insert column names record

Parameter: `add_item_names`

This option, when enabled, causes the Client to add a record with the DMSII item names to the start of all flat files. The names used are the item names in the DASDL, except for items that have OCCURS clauses. The latter get the suffix " *[n]*" appended to the name to make them unique, where *n* is the occurrence number of the item starting at 1. For example, an item named XXX with the clause "OCCURS 5 TIMES" will result in columns named XXX[1] through XXX[5].

Add new line characters to fixed format records

Parameter: `add_nl_chars`

This option, when enabled, causes the Client to automatically add new line characters to the end of fixed length records. The Client ignores this option when using variable length records, as they must end with new-line characters.

Enclose character data in quotes

Parameter: `use_quotes`

This option specifies whether or not character data should be enclosed in quotes. If this option is disabled, the **Quote Character** is ignored. This option is ignored when using the FIXED format.

Trim trailing blanks

Parameter: `rtrim_spaces`

This option controls whether or not the Client strips trailing blanks when using CSV format. COMMA format does not trim spaces, as it treats character data as fixed length. Furthermore, it does not do anything special with quote characters in the data that is enclosed in quotes, while CSV format uses two quote characters to represent a quote in the data (see the **Enclose character data in quotes** option above).

This option can be augmented by the **Set blank columns to NULL** parameter in the **CUSTOMIZING** page, which causes blank items to be represented as NULL (i.e. the empty string ",,").

Trim leading zeroes

Parameter: `ltrim_zeroes`

This option controls whether or not the Client strips leading zeroes for numbers in a CSV format.

Use + for positive signed numbers

Parameter: `use_plus_sign`

This option controls whether or not positive signed numbers are always preceded by a + sign.

Record Format

format

Parameter: `format`

Use this set of buttons to select the format for the records in the Flat Files created by the Client. The three supported formats are as follows:

- **CSV** format: this format vaguely resembles the VARYING FORMAT in DBSupport on the MCP that is used by DBSPAN. This format uses a delimiter (comma or tab character) to separate the fields in the record. Character data can optionally be enclosed in quotes.
- **Fixed** format: this is the equivalent of FIXED FORMAT in DBSupport on the MCP that is used by DBSPAN, where every item has a fixed length in the records.
- **Comma** format: this is the equivalent of COMMA FORMAT in DBSupport on the MCP that is used by DBSPAN.

Format DMSII REAL as

Parameter: `real_format`

Default: Scientific, 11, 6

Range: Scientific or Decimal

Configurator: PCSpan (Format DMSII REAL as)

This parameter specifies the format the Client uses on items whose DMSII data type is REAL. The two supported formats are as follows:

- **Scientific (E)** This format contains a normalized decimal number followed by an exponent preceded by the letter "E".
- **Decimal (F)** This format outputs without using an exponent.

Special Characters

COLUMN DELIMITER

Parameter: `delimiter`

These two options allow you to select the character that is used to separate columns in Flat Files created by the Client. This option is only meaningful when using CSV or COMMA formats, and defaults to ",". The other supported choice is the tab character.

DECIMAL POINT CHARACTER

Parameter: `decimal_char`

These two options allow you to select the decimal point character to be used in numeric items. It defaults to `"."` and the only other allowable value is `","` for European sites.

QUOTE CHARACTERS

Parameter: `quote`

These two options specify whether the Client will use the **Single Quote** or **Double Quote** character for quoted strings in Flat File records.

Directories

EXTRACTS

Parameter: `extracts_dir`

This option, when set to a non-blank value, defines the directory where the Client will create files containing extract records. If a directory is not specified, the files with extract records are written to the Client's working directory. By using the three separate directory specifications in this group you can make the Client use separate directories for extract, fixup, and update records. Typically you would set these to "extracts", "fixups" and "updates" to have separated directories.

FIXUPS

Parameter: `fixups_dir`

This option, when set to a non-blank value, defines the directory where the Client will create files containing fixup records. If a directory is not specified, the files with fixup records are written to the Client's working directory. By using the three separate directory specifications in this group you can make the Client use separate directories for extract, fixup, and update records.

Typically you would set these to "extracts", "fixups" and "updates" to have separated directories.

UPDATES

Parameter: `updates_dir`

This option, when set to a non-blank value, defines the directory where the Client will create files containing update records. If a directory is not specified, the files with update records are written to the Client's working directory. By using the three separate directory specifications in this group you can make the Client use separate directories for extract, fixup and update records.

Typically you would set these to "extracts", "fixups" and "updates" to have separated directories.

10.6.3 PCSpan - Advanced

This page contains advanced options for the Flat File Client.

Configuration file parameters are included below with the `following_font`.

Treat DMSII REAL As

Parameter: `treat_real_as`

This option indicates how the `define` and `redefine` commands treat all items that are not customized and have a DMSII data type of REAL. When enabled, no customizations are needed, unless you have exceptions that need to be treated differently. For an explantion of the 3 button options see the following:

- **Real:** indicates that a REAL will be treated as a floating point number
- **TIME(6):** indicates that a REAL will be treated as a TIME(6) value, which is a timestamp.
- **Binary:** indicates that a REAL will be written out to the flat files as 12-hex digits, just like an RSN.

If you have REAL items in DMSII that are timestamps, you need to treat them as BINARY (if they are TIME(6) data, treat them as such). If you just treat them as REAL, the most significant bit of the item will be lost when the number is converted to scientific or decimal formats, as this bit is unused for floating point values on the A-Series machines. On the other hand, timestamps use this bit, which is part of the date.

Dates

The options in this group define how dates are represented in flat files.

FLAT FILE DATE FORMATS

Parameter: `span_date_format`

This format uses the same date format codes as the standard Client. Currently, only the formats with 4-digit years are supported. Select the desired format from the provided list-box.

FLAT FILE DATE/TIME PRECISION (0 - 9)

Parameter: `span_date_scale`

This option specifies how many digits after seconds will be added to represent fractions of seconds. When the scale is greater than zero, a decimal character (dot or comma) is added after seconds.

FLAT FILE DATE DELIMITER

Parameter: `span_date_delim`

This option allows the year, month and day to either be separated by slashes, dashes, or dots (or not separated at all). You can select one of these choices from the list-box provided. When you use a date delimiter, the hours, minutes and seconds will be separated by colons and the time part will be preceded by a blank. If the delimiter is the empty string, then they are all run together to form a 14-digit number.

Prefix column mask

Parameter: `prefix_columns`

This option specifies a mask that is ANDed with the `default_user_columns` mask to determine which external columns are to be added to the start of the record (the remaining columns are added at the end of the record, like the standard Clients).

The bits defined for the parameter `default_user_columns` were expanded to include the items that DBSupport allows. Some of the columns used by the standard Clients are not useful or supported for the Flat File Client. Attempting to set them in the `default_user_columns` specification has no effect.

File name mask (0 minimum to 7 maximum)

Parameter: `filename_mask`

This option allows you to include any combination of the starting AFN, ABSN and TIMESTAMP to the fixups and updates flat file names. Once the filename suffix has been established it remains in effect for the duration of the run. The value is treated as a binary quantity where each bit mask represents an added component to the file names. A value of 1 indicates AFN, a value of 2 indicates ABSN and a value of 4 indicates TIMESTAMP.

A value of zero disables this option and the remaining values represent combinations of these three items. A value greater than 7 will be rejected as an error.

Post processing of flat files

When enabled, the next two entries become visible. Post processing is used in a special application of the Flat File Client that is used in conjunction with data redaction systems. The Client extracts the DMSII data sets to flat files, which are then passed on to the redaction system to anonymize sensitive information. The script workers are involved in interfacing the Client with the commercially available redaction software. When the redaction is complete, the new file gets passed back to the Client that runs a specialized Client application named **dbrebuild** converting the data in the flat file to its DMSII equivalent format with all ASCII data translated to EBCDIC.

The client will then launch Enterprise Server to transfer the file back to the MCP Server where a special utility loads the records into the test database that is being created by redacting the production data base, which allows for more realistic testing without compromising sensitive information.

SCRIPT WORKERS (1 MINIMUM TO 32 MAXIMUM)

Parameter: `n_pcspan_workers`

This option defines the number of script workers to be used in the post processing activity described above. This mechanism could be used for other applications.

ENABLE REDACTION

Parameter: `redacted_database`

When using the above mentioned redaction process this parameter must be enabled.

10.7 Encryption

Use the options on this page to configure encryption levels between the Client and the Databridge server on MCP systems.

Configuration file parameters are included below with the `following_font`.

10.7.1 Enable encryption for connections to the server

Parameter: `enable_encryption`

If the connection to the server is using TLS, enable this option. When this option is enabled enter the **CA path** and **CA file** information in the expanded input fields.

CA PATH

Parameter: `ca_path`

This input field specifies the fully qualified directory name where the certificate resides.

CA FILE

Parameter: `ca_file`

This input field specifies the fully qualified name of the file where the certificate resides.

10.7.2 Check server name in certificate

Parameter: `certify_server_name`

This option indicates that the Client should verify the server name in the certificate before it is accepted.

11. Legal Notice