EVIDEN

Identity and Access Management

DirX Audit

Command Line Interface Guide

Version 9.0, Edition July 2025

Table of Contents

Preface

This manual describes the command line interface provided with DirX Audit. It consists of the following chapters:

- Chapter 1 describes the command-line tools for working with the DirX Audit Database.

DirX Audit Documentation Set

DirX Audit provides a powerful set of documentation that helps you configure your audit server and its applications.

The DirX Audit document set consists of the following manuals:

- *DirX Audit Introduction*. Use this book to obtain a description of DirX Audit architecture and components.
- *DirX Audit Tutorial*. Use this book to get familiar quickly with your DirX Audit installation.
- *DirX Audit Administration Guide*. Use this book to understand the basic tasks of DirX Audit connectivity administration.
- *DirX Audit Manager Classic Guide*. Use this book to obtain a description of the DirX Audit Manager Classic user interface provided with DirX Audit.
- *DirX Audit Manager Guide*. Use this book to obtain a description of the DirX Audit Manager user interface provided with DirX Audit.
- *DirX Audit Command Line Interface Guide*. Use this book to obtain a description of the command line interface provided with DirX Audit.
- *DirX Audit Customization Guide*. Use this book to customize your DirX Audit environment.
- *DirX Audit History Synchronization Guide*. Use this book to obtain information about the DirX Audit History synchronization jobs.
- *DirX Audit Best Practices*. Use this book to obtain guidelines and tips for avoiding common mistakes and improving the experience of a DirX Audit installation.
- *DirX Audit Installation Guide*. Use this book to install DirX Audit.
- *DirX Audit Migration Guide*. Use this book to migrate DirX Audit from previous versions.
- *DirX Audit Release Notes*. Use this book to understand the features and limitations of the current release.
- *DirX Audit History of Changes*. Use this book to understand the features of previous releases.

Notation Conventions

Boldface type
In command syntax, bold words and characters represent commands or keywords that must be entered exactly as shown.

In examples, bold words and characters represent user input.

Italic type
In command syntax, italic words and characters represent placeholders for information that you must supply.

[ ]
In command syntax, square braces enclose optional items.

{ }
In command syntax, braces enclose a list from which you must choose one item.

In Tcl syntax, you must actually type in the braces, which will appear in boldface type.

|
In command syntax, the vertical bar separates items in a list of choices.

...
In command syntax, ellipses indicate that the previous item can be repeated.

install_path
The exact name of the root of the directory where DirX Audit programs and files are installed. The default installation directory is *userID_home_directory*/**DirX Audit** on UNIX systems and **C:\Program Files\DirX\Audit** on Windows systems. During installation the installation directory can be specified. In this manual, the installation-specific portion of pathnames is represented by the notation *install_path*.

install_media
The exact path where the DirX Audit installation media is located.

1. Using the DirX Audit Tools

DirX Audit provides the following database maintenance tools:

- A tool to maintain a database containing audit messages and history entries (DB maintenance)
- A tool to populate fact tables

The DB maintenance tool has the following modes:

- Export: Exports audit messages from the DirX Audit Data Database to XML files and exports history entries data from the DirX Audit History Database to JSON files. The audit messages can either be deleted or kept in the DirX Audit Data Database after export. The history entries data that was ended can either be deleted or kept in the DirX Audit History Database after export.
- Import: Imports audit messages from XML files to the DirX Audit Data Database. The XML files can be either deleted or kept in the file system after import.
- Extend: Extends audit messages in the DirX Audit Data Database with audit events and dimensions.
- Compress: Compresses original messages in the DirX Audit Data Database.
- Purge: Purges audit messages data and history entries data from the DirX Audit Database.
- Remove duplicate: Removes duplicate history entries with the same dirxEntryUUID or DN or duplicate history entries linked to the same logical entry.
- Compute context: computes missing contexts for audit events within the specified time interval.

The following sections provide information about these tools.

1.1. General Information

This section provides information common to all the DirX Audit tools, including:

- Usage prerequisites
- Installation location
- Common syntax and options

1.1.1. Usage Prerequisites

The DirX Audit tools have the following prerequisites:

- A command shell (the Windows command prompt or a UNIX shell) must be available to run the tools.
- The Java Virtual Machine (JVM) must be set up correctly. You must use the same JVM version that the other DirX Audit components use. If the tool is installed from the DirX Audit installer, the correct JVM will be used automatically. When the tool is installed from a standalone package, you must set the system path to contain the **bin** folder of the JVM and set the **JAVA_HOME** system environment variable to point to the JVM folder.

1.1.2. Installation Location

The DirX Audit tools are located in sub-folders of the folder:

*install_path***/tools/***tool_identifier*

where

tool_identifier corresponds to the tool as follows:
db_maintenance indicates the DB maintenance tool.
db_fact_population indicates the fact population tool.

The sub-folder **bin** contains the binary of the tool.

The names of the binaries are:

- For the DB maintenance tool:
  dxtdbtool.bat on Windows and **dxtdbtool.sh** on UNIX.
  It is referred to as **dxtdbtool** for the rest of this chapter.

  dxthistdbtool.bat on Windows and **dxthistdbtool.sh** on UNIX.
  It is referred to as **dxthistdbtool** for the rest of this chapter.
- For the fact population tool:
  dxtPopulateFacts.bat on Windows and **dxtPopulateFacts.sh** on Unix.
  It is referred as **dxtPopulateFacts** for the rest of this chapter.

1.1.3. Common Syntax

The general usage of the tools is:

tool_name command [*options*]

where

tool_name
is either **dxtdbtool** or **dxthistdbtool** for the DB maintenance tool or **dxtPopulateFacts** for the fact population tool.

command
is one of the following keywords:

- **help** or **export** or **import** or **extend** or **compress|decompress** or **purge** or **computeContext** for **dxtdbtool**
- **help** or **purge** or **purgeorphans** or **purgeended** or **export** or **remdup** or **ldapremdup** or **filluuid** or **makeunique** for **dxthistdbtool**
- no command for **dxtPopulateFacts**

options
is a list of common and command specific options. (See the following sections for details.)

If a value contains a SPACE character enclose the value in double quotes (").

You can always run the tool without any argument (not **dxtPopulateFacts**) or use *`tool_name`* `help` to get the usage help information and examples. This help is always up to date for your installed version of DirX Audit, in case of differences with this document, follow the help directly in the command line tools.

1.1.4. Common Options

All DirX Audit tools recognize the following options:

-debug
Creates a debug log.

-debugMemory
Displays more detailed memory allocation information.

-silent
Suppresses the user confirmation dialog for performing a given action. By default, you must confirm (by entering **yes**) that you really want to perform an action. This option allows you to bypass the confirmation dialog.

-simulate
Shows the results of an action without actually performing the action.

-tenantid *tenantID*
Specifies the tenant whose configuration file is accessed to acquire database credentials. The *tenantID* specifies the identifier of a configured tenant.

-transize *tranSize*
Specifies the transaction size used during an operation. A larger size will increase performance, but will require larger transaction log files.

-types *type*[,*type*]...
Specifies a list of comma-separated history entry types to which the operation is restricted. All types are used, if none is specified.

The tenant ID is used to obtain the database credentials of the specific tenant. The credentials are acquired from the tenant configuration file, properly configured by the Configuration Wizard. If only one tenant is configured, the **-tenantid** option can be omitted from the command line and the configuration file of this tenant is used automatically. If more than one tenant is configured, the **-tenantid** option needs to be specified on the command line.

1.1.5. LDAP Connection Parameters

Some DirX Audit tools require the following LDAP connection parameters:

-ldapbase *ldapBase*
LDAP node used as a search base.

-ldapconfig *ldapConfigFile*
LDAP connection properties file. It contains all mentioned required LDAP parameters. You can find the **ldap.properties** file example in the *install_path***/tools/db_maintenance/doc/samples**.

-ldapdomain *ldapDomain*
LDAP domain DN. It will be prepended before the *ldapBase* node.

-ldapfilter *ldapFilter*
Search filter applied to the LDAP search.

-ldaphost *ldapHost*
LDAP host used to establish an LDAP connection.

-ldappassword *ldapPassword*
LDAP connection password.

-ldapport *ldapPort*
Port on which the LDAP connection will be established.

-ldapscope *ldapScope*
Scope of the LDAP search.

-ldapuser *ldapUser*
LDAP user used to establish an LDAP connection.

-ldapusessl *ldapUseSSL*
Specifies whether to use SSL for the LDAP connection.

-ldaptruststorepath *ldapTruststorePath*
Path to the truststore file used for the LDAP connection.

-ldaptruststorepassword *ldapTruststorePassword*
Truststore password used for the LDAP connection.

1.2. Maintaining Audit Messages

This section describes how to use the DB maintenance tool to maintain DirX Audit messages.

1.2.1. Export Audit Trail

To export audit messages from the DirX Audit Data Database to XML files, execute the following command:

dxtdbtool export *common_options*
-dstdir *folder_name*
[**-delete**]
[**-from** *date_time | function*]
[**-recsperfile** *number*]
[**-recsperquery** *number*]
[**-to** *date_time | function*]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-dstdir *folder_name*
Specifies the destination folder for the XML files containing the exported audit messages.

-delete
Specifies that the audit messages are to be deleted from the DirX Audit Data Database after export.

-from *date_time | function*
Specifies that only audit messages created after the specified date and time (inclusive) or by the specified function are exported.

-recsperfile *number*
Specifies the maximum number of audit messages to be written to the XML files.
The default value is **500**.

-recsperquery *number*
Specifies the maximum number of audit messages per database query. The value can affect command processing on the database-server side. A low value can increase the total processing time because of frequent network communication and more actions to be processed. On the other hand, a high value can require more database server resources and command processing can fail when these resources are not sufficient.
The default value is **500**. Decrease this number if the export fails.

-to *date_time | function*
Specifies that only audit messages created up to the specified date and time (exclusive) or by the specified function are exported.

The format for the value of *date_time* is *yyyy.MM.dd_HH.mm.ss_z* . See the Java documentation for the SimpleDateFormat class for details and examples.

You can use the following functions in **-from** and **-to** options:

- Beginning functions (which return the beginning of a unit): **$bhour(***parameter***)**, **$bday(***parameter***)**, **$bweek(***parameter***)**, **$bmonth(***parameter***)**, **$byear(***parameter***)**
- End functions (which return the end of a unit): **$ehour(***parameter***)**, **$eday(***parameter***)**, **$eweek(***parameter***)**, **$emonth(***parameter***)**, **$eyear(***parameter***)**
- Other: **$now()**

The *parameter* argument is a required integer value, where:

- Zero (0) represents the current moment. For example, **$bday(0)** returns the beginning of this day.
- A negative number represents a moment before the current date and time value. For example, **$emonth(-1)** returns the end of the previous month.
- A positive number represents a moment after the current date and time value. For example, **$byear(3)** returns the beginning of a year in three years.

Note that audit messages usually use UTC zone in the audit message **Identification - When** field.

If the **-delete** option has been specified and an error occurs during export or the tool is interrupted, it can resume work: simply re-run the tool with exactly the same arguments.

1.2.1.1. Exported Message Structure and Format

The exported audit messages are stored in the file system as XML files. The files that contain exported messages are compressed. The audit messages are split into separate files using the audit message event date and given options according to the following rules:

- A new folder structure *yyyy*/*MM* is created in the destination folder (option **-dstdir**) for each file set if it does not exist yet; for example: the folder **2010/01** will contain messages from January 2010.
- The **-recsperfile** option specifies the maximum number of audit messages in a file.
- One file contains audit messages from one day. (The audit message **Identification -**

When field is used.)

The file containing exported audit messages has the following name and path:

*yyyy/MM***/dxtdata_***yyyyMMdd_HHmm-HHmm***_tot***records_count_index***.xml.zip**

Here is an example:

```
2001/12/dxtdata_20011217_0930-0930_tot1_0.xml.zip
```

An additional file with the same base name and suffix **.info.xml** is created for each exported messages file. This file contains information about the exported messages.

1.2.1.2. Export Examples

In the following example, all messages from January 2009 are exported. The export process deletes the exported messages from the DirX Audit Data Database. It stores the data into the folder **./data/audit_export/**. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**. Please make sure that you use the "simple plain hyphen" **-** in the command line.

```
dxtdbtool export -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -dstdir
./data/audit_export -from 2009.01.01_00.00.00_UTC -to
2009.02.01_00.00.00_UTC -delete
```

In the following example, all messages from January 2009 are exported. The export process deletes the exported messages from the DirX Audit Data Database. It stores the data into the folder **./data/audit_export/**. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID):

```
dxtdbtool export -dstdir ./data/audit_export -from 2009.01.01_00.00.00_UTC
-to 2009.02.01_00.00.00_UTC -delete
```

In the following example, all messages older than 12 months are exported. The export process deletes the exported messages from the DirX Audit Data Database. It stores the data in the folder **./data/audit_export/**. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxtdbtool export -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -dstdir
./data/audit_export -to $bday(-365) -delete
```

In the next example, all messages are exported. The export process does not delete the exported messages from the DirX Audit Data Database. It stores the data in the folder **./data/audit_export**. The database connection data are taken from the stored configuration file of the only configured tenant (set in the Configuration Wizard). If multiple tenants are configured you must specify the tenant ID. The export operation starts immediately (because the **-silent** option is used; see the section "Common Options" for details):

```
dxtdbtool export -dstdir ./data/audit_export -silent
```

1.2.2. Import Audit Trail

To import audit messages from XML files to the DirX Audit Data Database, execute the following command:

dxtdbtool import *common_options*
-src *path*
[**-delete** | **-dstdir** *folder_name*]
[**-recursive**]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-src *path*
Specifies the source file or folder. You can also use a folder containing data previously exported with **dxtdbtool export** saved as **dxtdata_***xxx***.xml.zip**.

-delete
Specifies that the XML files are deleted after import.

-dstdir *folder_name*
Specifies the destination folder to which the XML files are moved after import.

-recursive
Specifies that the XML files in all sub-folders are imported.

The options **-delete** and **-dstdir** are mutually exclusive. You can specify only one of them. After a successful import, the source file is either deleted from the file system (if **-delete** was specified) or moved from the source folder into destination folder (if **-dstdir** was specified).

The option **-src** can be either a path to a file or a folder. If it is a folder, only the files in this folder are processed unless the **-recursive** option was specified (all sub-folders and files are processed in this case).

The import operation expects the files to be in the same format and syntax as generated by the export operation; that is, in the form of archive files.

If an error occurs during the import process or the tool is interrupted, it can resume the import: simply re-run the tool with exactly the same arguments.

You can also use the DirX Audit file collector component of the DirX Audit Server to import the messages from the XML files into the DirX Audit Data Database. However, you must unzip all audit message files before you copy them into the collector's input folder. Do not copy the information files there. Only the audit message files should be copied to this folder. See the section "Server File Collector for DirX Audit Format" in the *DirX Audit Installation Guide* for the configuration details.

1.2.2.1. Import Examples

In the following example, the audit messages are imported from the folder **./data/audit** to the DirX Audit Data Database. All connection settings are taken from the stored tenant configuration file. After import, the files are moved to the **./data/audit/archive**. Please make sure that you use the "simple plain hyphen" **-** in the command line.

```
dxtdbtool import -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -src
./data/audit -dstdir ./data/audit/archive -recursive
```

In the next example, the audit messages are imported from the folder **./data/audit** to the DirX Audit Data Database. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID). After import, the files are deleted:

```
dxtdbtool import -src ./data/audit -delete -recursive
```

In the next example, the audit messages from the file/archive **./data/audit/file.xml.zip** are imported to the DirX Audit Data Database. The connection settings are taken from the stored configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID). After import, the file is moved to the folder **./data/audit_backup**:

```
dxtdbtool import -src ./data/audit/file.xml.zip -dstdir ./data/audit_backup
```

1.2.3. Extend Audit Messages

To extend audit messages in the DirX Audit Data Database with audit events and dimensions, execute the following command:

dxtdbtool extend *common_options*
[**-from** *date_time*]
[**-recsperquery** *number*]
[**-to** *date_time*]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-from *date_time*
Specifies that only audit messages created after the specified date and time (inclusive) are extended.

-recsperquery *number*
Specifies the maximum number of audit messages per database query. The value can affect command processing on the database-server side. A low value can increase the total processing time because of frequent network communication and more actions to be processed. On the other hand, a high value can require more database server resources

and command processing can fail when these resources are not sufficient. The default value is **500**. Decrease this number if the command fails.

-to *date_time*
Specifies that only audit messages created up to the specified date and time (exclusive) are extended.

The format for the value of *date_time* is *yyyy.MM.dd_HH.mm.ss_***z**. See the Java documentation for the SimpleDateFormat class for details and examples.

Note that audit messages usually use UTC zone in the audit message **Identification - When** field.

Extending audit messages is not usually necessary, especially if you use the import tool. It is only necessary in specific cases and is then indicated in the appropriate places; in the *DirX Audit Release Notes*, for example.

1.2.3.1. Extend Examples

In the following example, all messages are extended. The connection data are stored in the configuration file of the tenant with the ID **4f753e1d-d0de-4aef-bb22-caace7342e99**. Make sure that you use the "simple plain hyphen" **-** in the command line.

```
dxtdbtool extend -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
```

In the next example, all messages from January 2019 are extended. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxtdbtool extend -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -from
2019.01.01_00.00.00_UTC -to 2019.02.01_00.00.00_UTC
```

In the next example, all messages are extended. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID):

```
dxtdbtool extend
```

1.2.4. Compress and Decompress Original Message

Original messages can be optionally stored together with audit messages. In DirX Audit 7.0 SP1 and earlier versions, they were saved as text. As of DirX Audit 7.1, they can be saved in a compressed form, which can significantly reduce the database size.

The original message in the text form is saved in the ORIGINALMESSAGE column of the DAT_ORIGINALMESSAGES table. The original message in the compressed form is saved in the ORIGINALMESSAGE_COMPRESS column of the same table.

The command compresses the original message text data and stores it in the compressed form. It removes the text data from the table. When the command is completed, the table's clustered index is rebuilt. Data space should be significantly reduced. The table size should be reduced to about 20 % of its previous size.

To compress or decompress original messages in the DirX Audit Data Database, execute the following command:

dxtdbtool compress|decompress *common_options*
[**-from** *date_time*]
[**-recsperquery** *number*]
[**-to** *date_time*]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-from *date_time*
Specifies that only audit messages created after the specified date and time (inclusive) are compressed.

-recsperquery *number*
Specifies the maximum number of audit messages per database query. The value can affect command processing on the database-server side. A low value can increase the total processing time because of frequent network communication and more actions to be processed. On the other hand, a high value can require more database server resources and command processing can fail when these resources are not sufficient. The default value is **500**. Decrease this number if the command fails.

-to *date_time*
Specifies that only audit messages created up to the specified date and time (exclusive) are compressed.

The format for the value of *date_time* is *yyyy.MM.dd_HH.mm.ss_***z**. See the Java documentation for the SimpleDateFormat class for details and examples.

Note that audit messages usually use UTC zone in the audit message **Identification - When** field.

1.2.4.1. Compress and Decompress Examples

In the following example, all original messages are compressed. The connection data are stored in the configuration file of the tenant with the ID **4f753e1d-d0de-4aef-bb22-caace7342e99**. Make sure that you use the "simple plain hyphen" **-** in the command line.

```
dxtdbtool compress -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
```

To decompress, use the following command:

```
dxtdbtool decompress -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
```

In the next example, all messages from January 2019 are compressed. The connection data are stored in the configuration file of the tenant with the ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxtdbtool compress -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -from
2019.01.01_00.00.00_UTC -to 2019.02.01_00.00.00_UTC
```

To decompress, use the following command:

```
dxtdbtool decompress -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -from
2019.01.01_00.00.00_UTC -to 2019.02.01_00.00.00_UTC
```

In the next example, all messages are compressed. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID):

```
dxtdbtool compress
```

To decompress, use the following command:

```
dxtdbtool decompress
```

1.2.5. Purge Audit Messages Data

To purge audit messages data from the DirX Audit Data Database, execute the following command:

dxtdbtool purge *common_options*
-scope {**DAT_AUDITMESSAGES** | **DAT_AUDITMESSAGES_ADDITIONS** | **DAT_ORIGINALMESSAGES**}
-filter *file_path*
[**-paramsfile** *file_path*]
[**-params** *key*[**.DATETIME]=***value*[,*key*[**.DATETIME]=***value*]...]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-scope {**DAT_AUDITMESSAGES** | **DAT_AUDITMESSAGE_ADDITIONS** | **DAT_ORIGINALMESSAGES**}
Specifies the scope of the audit message data to be purged with a table name:

- **DAT_AUDITMESSAGES** deletes complete audit messages including message additions and original messages.
- **DAT_AUDITMESSAGE_ADDITIONS** deletes only message additions and original messages, but it keeps **DAT_AUDITMESSAGES** content.
- **DAT_ORIGINALMESSAGES** deletes only original messages.

-paramsfile *file_path*
Specifies a path to the file containing a list of SQL select statement parameters and their values in the format *key*=*value* for common types and *key*.**DATETIME=***value* for the date and time type.
The **-paramsfile** parameter can be omitted when no SQL select statement parameter is used or when only inline parameters are provided. When a parameter key is used both in the file and inline, the inline value takes precedence.

-params *key*[**.DATETIME]=***value*[,*key*[**.DATETIME]=***value*]...
Represents a set of SQL select statement parameter key – value pairs. Optionally, the parameter can be of date and time data type.

The format for the value of *date_time* is **yyyy.MM.dd_HH.mm.ss_z**. See the Java documentation for the SimpleDateFormat class for details and examples.

You can use the same functions as described in the section "Export Audit Trail" to specify values of date and time parameters (dateparam).

Note that audit messages usually use UTC zone in the audit message **Identification - When** field.

1.2.5.1. Purge Audit Data Examples

In the following example, original messages are removed. The connection data are stored in the configuration file of the tenant with the ID **4f753e1d-d0de-4aef-bb22-caace7342e99**. Make sure that you use the "simple plain hyphen" **-** in the command line.

```
dxtdbtool purge
-tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-scope DAT_ORIGINALMESSAGES
-params "IDENTIFICATION_SOURCE=DirX Identity",
IDENTIFICATION_OUTCOME=0,
WHEN_FROM.DATETIME=$bmonth(-6),
WHEN_TO.DATETIME=$bmonth(-5)
```

The same result can be achieved with a parameter file:

```
dxtdbtool purge
-tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-scope DAT_ORIGINALMESSAGES
-paramsfile om_params.properties
```

The content of the **om_params.properties** file can look like this:

```
IDENTIFICATION_SOURCE=DirX Identity
IDENTIFICATION_OUTCOME=0
WHEN_FROM.DATETIME=$bmonth(-6)
WHEN_TO.DATETIME=$bmonth(-5)
```

1.2.6. Compute Audit Event Context

These options allows computing missing audit events context data independently from the similar server job.

To generate missing audit events context in the DirX Audit Data Database, execute the following command:

dxtdbtool computeContext *common_options*
[**-from** *date_time*]
[**-to** *date_time*]
[**-childrenlinkingbatch** *number*]
[**-maxresult** *number*]
[**-ordering** *string*]
[**-orphanprocessingbatch** *number*]
[**-orphanto** *date_time*]

where

common_options
Specifies the common options. See the section “Common Options” for details.

-from *date_time*
Specifies that context is computed for audit messages created after the specified date and time (inclusive).

-to *date_time*
Specifies that context is computed only for audit messages created up to the specified date and time (exclusive).

-childrenlinkingbatch *number*
Batch size for linking children to parents. All children are always processed within a context processing iteration. This parameter only influences how many children are processed at once in a single batch. Default is 20000 for MSSQL and 1000 for ORACLE.

-maxresult *number*
Batch size per iteration. If an iteration lasts too long, lower the number.

-ordering *string*
Specifies the ordering of the context calculation. Possible values are **ASC|DESC**. **DESC**

means, that contexts will be calculated from the newest messages to the oldest. The default value is **ASC**.

-orphanprocessingbatch *number*
Specifies the batch size for processing orphaned messages. This parameter determines how many orphaned messages are processed in a single context processing iteration. The default value is 5000.

-orphanto *date_time*
Specifies that orphaned messages older than the specified date and time (exclusive) are processed. It allow to handle old messages, that could not be associated with a context because they are older than the specified date and time.

1.2.6.1. Compute Audit Event Context Examples

In the following example, missing context records will be generated for events from January 2009 till March 2009. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID):

```
dxtdbtool computeContext -from 2009.01.01_00.00.00_UTC -to
2009.04.01_00.00.00_UTC
```

In the next example, missing context records will be generated for specific tenant and events from now up to 6 months ago. The connection data are stored in the configuration file of the tenant with ID **27998a9d-d518-40c1-92c8-636573608c28**:

```
dxtdbtool.bat computeContext -tenantid 27998a9d-d518-40c1-92c8-636573608c28
-from $bmonth(-6) -orphanto $bmonth(-6)
```

1.3. Maintaining History Entries

This section describes how to use the DB maintenance tool to maintain history entries.

1.3.1. Purge History Entries Data

The DB maintenance tool can delete history entries data from the DirX Audit History Database that have already ended. Specifically, it can remove rows of the following tables according to their VALID_TO column value:

- HST_ENTRIES_IN_TIME
- HST_SMALL_ATTRS_IN_TIME
- HST_LINK_ATTRS_IN_TIME
- HST_LARGE_ATTRS_IN_TIME
- HST_ROLEPARAMS_IN_TIME
- HDB_SMALL_DATTRS_IN_TIME
- HDB_LINK_DATTRS_IN_TIME
- HDB_MANUAL_ASSIGNMENTS

To purge history entries data from the DirX Audit History Database, execute the following command:

dxthistdbtool purge *common_options*
-endedbefore *date_time* | *function*
[**-endedafter** *date_time* | *function*]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-endedbefore *date_time* | *function*
Specifies that only history entries data ended before the specified date and time (exclusive) or by the specified function are deleted.

-endedafter *date_time* | *function*
Specifies that only history entries data ended after the specified date and time (inclusive) or by the specified function are deleted.

The format for the value of *date_time* is *yyyy.MM.dd_HH.mm.ss_z*. See the Java documentation for the SimpleDateFormat class for details and examples.

You can use any of the functions described in the section "Export Audit Trail" in the **-from** and **-to** options.

1.3.1.1. Purge History Data Examples

In the following example, all history entries data ended in January 2019 are deleted. The purge process deletes the history entries data from the DirX Audit History Database. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**. Make sure that you use the "simple plain hyphen" **-** in the command line.

```
dxthistdbtool purge -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-endedafter 2019.01.01_00.00.00_UTC -endedbefore 2019.02.01_00.00.00_UTC
```

In the next example, all history entries data ended in January 2019 are deleted. The purge process deletes the history entries data from the DirX Audit History Database. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID):

```
dxthistdbtool purge -endedafter 2019.01.01_00.00.00_UTC -endedbefore
2019.02.01_00.00.00_UTC
```

In the next example, all history entries data ended before 6 months (end of months) are deleted. The purge process deletes the history entries data from the DirX Audit History Database. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool purge -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-endedbefore $bmonth(-5)
```

In this example, all history entries data ended before the end of the last year are deleted. The purge process deletes the history entries data from the DirX Audit History Database. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool purge -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-endedbefore $byear()
```

In the next example, all history entries data ended during the previous month are deleted. The purge process deletes the history entries data from the DirX Audit History Database. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool purge -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-endedafter $bmonth(-1) -endedbefore $bmonth()
```

In the next example, all history entries data ended before today are deleted. The purge process deletes the history entries data from the DirX Audit History Database. The connection data are stored in the configuration file of the tenant with **ID 4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool purge -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-endedbefore $bday()
```

In the last example, all ended history entries data are deleted. The purge process deletes the history entries data from the DirX Audit History Database. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID). The purge process starts immediately because the **-silent** option is used; see the section "Common Options" for details:

```
dxthistdbtool purge -silent
```

1.3.2. Purge Orphaned History Entries

Purging history entries data using the purge tool can leave so-called “orphaned” entries. Orphaned entries are entries that have no corresponding attributes. Purging those orphans can be performed by executing the following command:

dxthistdbtool purgeorphans *common_options*
[-forcedelete]

where

common_options
Specifies the common options. See the section “Common Options” for details.

-forcedelete
In case of data inconsistencies in the database, some orphaned entries can have lingering attributes which fail the purge. This flag forces the deletion of orphaned entries together with any would-be lingering attributes.

1.3.2.1. Purge Orphaned History Entries Examples

In the following example, all orphaned history entries are purged from the database. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99:**

```
dxthistdbtool purgeorphans -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
```

In the next example, all orphaned history entries present in the database of types 'Account' and 'User' are purged from the database. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool purgeorphans -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-types "Account,User"
```

In the next example, all orphaned history entries are purged from the database. In case of any lingering attributes linked to the orphans cause by database inconsistencies, the orphans are purged along with the lingering attributes. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool purgeorphans -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-forcedelete
```

1.3.3. Purge Ended History Entries

Another approach to purging history entries is by purging ended entries. Ended entries are identified as entries, that have no valid attributes. As opposed to the **purge** tool, which only purged the ended attributes of entries, this approach purges the entire entries along with their attributes. Purging ended entries can be performed by executing the following command:

dxthistdbtool purgeended *common_options*
-endedbefore *date_time | function*

where

common_options
Specifies the common options. See the section "Common Options" for details.

-endedbefore *date_time | function*
Purge ended history entries whose validity ended before this date and time (exclusive).

1.3.3.1. Purge Ended History Entries Examples

In the following example, all ended history entries are purged from the database. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99:**

```
dxthistdbtool purgeended -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
```

In the next example, all ended history entries, whose last validity ended before the specified date and time. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool purgeended -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-endedbefore $bmonth(-1)
```

1.3.4. Export History Entries

To export history entries data from the DirX Audit History Database to JSON files, execute the following command:

dxthistdbtool export *common_options*
-dstdir *folder_name*
[**-delete**]
[**-includestarted**]
[**-endedafter** *date_time | function*]
[**-endedbefore** *date_time | function*]
[**-entsperfile** *number*]
[**-nouidlist**]
[**-noorphans**]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-dstdir *folder_name*
Specifies the destination folder for the JSON files containing the exported history entries data.

-delete
Specifies that the history entries data that were ended in given date-time range are to be purged from the DirX Audit History Database after export.
A date and time range must be defined at least by the -**endedbefore** option. The default is not to delete history entries data. Started entries and attributes that have not ended are not affected.

-includestarted
Includes history entries data that were started in the given date and time range into the export process. The default exports only ended history entries data in the given date and time range.

-endedafter *date_time* | *function*
Exports only those history entries data ended after the specified date and time (inclusive) or by the specified function.

-endedbefore *date_time* | *function*
Exports only those history entries data ended up to the specified date and time (exclusive) or by the specified function.

-entsperfile *number*
Specifies the maximum number of history entries represented by JSON files to be written to the ZIP files. The default value is 500.

-nouidlist
Specifies that lists of entries identifiers will not be created. The default is to create such a list for every type folder.

-noorphans
Specifies that assignments without references to link attributes will not be exported. The default is to export such assignments into a separate folder.

The format for the value of *date_time* is *yyyy.MM.dd_HH.mm.ss_z*. See the Java documentation for the SimpleDateFormat class for details and examples.

1.3.4.1. Exported Entries Structure and Format

The exported history entries data are stored in the file system as JSON files. These files are compressed by the number specified in the **-entsperfile** option. The folder structure is as follows:

- A new folder structure **dxthistory_***yyyyMMddHHmmss/type* is created in the destination folder (option **-dstdir**) for each exported entry type; for example, the folder **`dxthistory_20220911093110/Account`** will contain all history entries data according to the given date and time range. Date and time in the name of top folder is the timestamp of the time at which the export started.
- The **-entsperfile** option specifies the maximum number of history entries in ZIP file.

The file containing exported history entries data has the following name and path structure:

dxthistory_*yyyyMMddHHmmss/type/type_count_range*.**zip**

Here is an example:

`dxthistory_20220911093110/Account/Account_000000001_000000500.zip`

Each type folder contains a **dxruid_list.txt** file, which contains a list of entries identifiers for a specific type.

The additional files **export_info.txt**, **domain.properties**, **parameters.properties**, **export_result_success** or **export_result_error** are created in the folder **dxthistory**_*yyyyMMddHHmmss*. When the **-delete** option is used, additional files such as **purge_info.txt**, **purge_result_success** or **purge_result_error** are also created. The **info** file contains information about the amount of exported/purged data and the operation's duration. After the export is finished, the common **result_success** or **result_error** is created.

1.3.4.2. History Export Examples

In the following example, all history entries data that ended in January 2009 are exported. The export process purges exported history entries data from the DirX Audit History Database. It stores the data into the folder **./data/history_export/**. The connection data are stored in the configuration file of the tenant with **ID 4f753e1d-d0de-4aef-bb22-caace7342e99**.
Please make sure that you use the "simple plain hyphen" - in the command line.

```
dxthistdbtool export -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -dstdir
./data/history_export -endedafter 2009.01.01_00.00.00_UTC -endedbefore
2009.02.01_00.00.00_UTC -delete
```

In the next example, all history entries data that ended in January 2009 are exported. The export process purges exported history entries data from the DirX Audit History Database. It stores the data into the folder **./data/history_export/**. The connection data are stored in the configuration file of the only configured tenant (if multiple tenants are configured you must specify the tenant ID):

```
dxthistdbtool export -dstdir ./data/history_export -endedafter
2009.01.01_00.00.00_UTC -endedbefore 2009.02.01_00.00.00_UTC -delete
```

In this example, all history entries data that ended previous month are exported. The export process purges exported history entries data from the DirX Audit History Database. It stores the data into the folder **./data/history_export/**.
The connection data are stored in the configuration file of the tenant with **ID 4f753e1d-d0de-4aef-bb22-caace7342e99**.

```
dxthistdbtool export -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -dstdir
./data/history_export -endedafter $bmonth(-1) -endedbefore $bmonth() -delete
```

In the following example, all history entries data that started or ended up to now are

exported. It stores the data into the folder **./data/history_export/**. The connection data are stored in the configuration file of the tenant with **ID 4f753e1d-d0de-4aef-bb22-caace7342e99**.

```
dxthistdbtool export -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -dstdir
./data/history_export -includestarted
```

In the next example, Account and User history entries data that ended up to one year ago are exported. The export process purges the exported history entries data from the DirX Audit History Database. It stores the data in the folder **./data/history_export/**. The connection data are stored in the configuration file of the tenant with ID **4f753e1d-d0de-4aef-bb22-caace7342e99**:

```
dxthistdbtool export -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -dstdir
./data/history_export -endedbefore $bday(-365) -types "Account,User" -delete
```

In the last example, all history entries data that have started or ended up to now are exported. The export process does not purge the exported history entries data from the DirX Audit History Database. It stores the data in the folder **./data/history_export/**. The database connection data are taken from the stored configuration file of the only configured tenant (set in the Configuration Wizard). (If multiple tenants are configured you must specify the tenant ID). The export operation starts immediately (because the **-silent** option is used; see the section "Common Options" for details):

```
dxthistdbtool export -dstdir ./data/history_export -includestarted -silent
```

1.3.5. Remove Duplicate History Entries

To remove duplicate history entries data from the DirX Audit History Database, execute the following command:

dxthistdbtool remdup *common_options*
-searchby {**ENTRY** | **DIRX_ENTRY_UUID** | **DN**}

where

common_options
Specifies the common options. See the section "Common Options" for details.

-searchby {**ENTRY** | **DIRX_ENTRY_UUID** | **DN**}
Specifies whether the duplicate entries should be identified by entry, dirxEntryUUID (default) or DN:

- **ENTRY** - Finds logical entries with multiple current history entries and ends all HST_ENTRIES_IN_TIME records associated to the same logical entry except the last created one.
- **DIRX_ENTRY_UUID** - Finds history entries with duplicate dirxEntryUUID values and ends the associated HST_ENTRIES_IN_TIME records except the last created one.

- **DN** - Finds history entries with multiple HST_ENTRIES_IN_TIME records that are not ended and ends all records associated to each entry except the last created one.

1.3.5.1. Remove Duplicate History Entries Examples

In this example, the tool finds all history entries linked to the same logical entry and ends all associated history entry records except the last one created. The connection data are stored in the configuration file of the tenant with ID **fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0**.

```
dxthistdbtool remdup -tenantid fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0
-searchby ENTRY
```

In the next example, the tool finds all history entries with duplicate dirxEntryUUID values and ends associated history entry records except the last one created. The connection data are stored in the configuration file of the tenant with ID **fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0**.

```
dxthistdbtool remdup -tenantid fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0
-searchby DIRX_ENTRY_UUID
```

In this example, the tool finds all history entries with multiple records having the identical DN value that are not ended and ends all records associated to each entry except the last one created of the only configured tenant (if multiple tenants are configured you must specify the tenant ID).

```
dxthistdbtool remdup -searchby DN
```

1.3.6. Remove Duplicate LDAP Entries

To remove entries with duplicate dirxEntryUUID attributes from LDAP, and optionally synchronize the changes made in LDAP to the DirX Audit History Database, execute the following command:

dxthistdbtool ldapremdup *common_options ldap_parameters*
[**-filluuid**]
[**-showonly**]
[**-synctohistdb**]

where

common_options
Specifies the common options. See the section “Common Options” for details.

ldap_parameters
Specifies LDAP connection parameters. See the section “LDAP Connection Parameters” for details.

-filluuid
If entries with missing dirxEntryUUID attribute are found, they will have new values generated. By default, they are ignored.

-showonly
Only shows the empty or duplicated values for dirxEntryUUID attribute and DN values of corresponding entries.

-synctohistdb
Synchronize the changes made in LDAP to DirX Audit History Database. Therefore, if a duplicate dirxEntryUUID is changed in LDAP, the same action occurs in the DirX Audit History Database if the corresponding entry exists. Entries in LDAP and DirX Audit History Database are determined to be corresponding if they match either on dirxEntryUUID/DN or dirxEntryUUID/dxrUID value pairs.

1.3.6.1. Remove Duplicate LDAP Entries Examples

In this example, entries with duplicate dirxEntryUUID values are retrieved from LDAP. Duplicate dirxEntryUUIDs have new values generated in LDAP. Entries with missing dirxEntryUUID values are skipped. This is the default behavior when no options are selected.

```
dxthistdbtool ldapremdup -ldapconfig ldap.properties
```

In this example, the tool searches LDAP for entries with duplicate or missing dirxEntryUUID values and prints them out. No modifications are made.

```
dxthistdbtool ldapremdup -ldapconfig ldap.properties -showonly
```

In the next example, the behavior in LDAP stays the same as in the default case (the first example); however, the changes made in LDAP are synchronized to the DirX Audit History Database for any matching entries. The connection data for the LDAP are stored in the ldap.properties file. The connection data for the tenant are stored in the configuration file of the tenant with ID **fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0**.

```
dxthistdbtool ldapremdup -ldapconfig ldap.properties -synctohistdb -tenantid
fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0
```

In the following example, the tool searches for and solves duplicated dirxEntryUUIDs and searches for missing dirxEntryUUIDs. If LDAP entries with missing dirxEntryUUID values are found, a new dirxEntryUUID value is generated for them in LDAP.

```
dxthistdbtool ldapremdup -ldapconfig ldap.properties -filluuid
```

1.3.7. Fill Missing dirxEntryUUID Values of History Entries

To generate new unique values for missing dirxEntryUUID attributes of history entries in DirX Audit History Database, execute the following command. No modifications are made in LDAP.

dxthistdbtool filluuid *common_options ldap_parameters*
[**-uuidfiltersize** *size*]

where

common_options
Specifies the common options. See the section “Common Options” for details.

ldap_parameters
Specifies LDAP connection parameters. See the section “LDAP Connection Parameters” for details. If an entry with a missing dirxEntryUUID value is found in the DirX Audit History Database, the tool searches LDAP for a matching entry and if possible, uses this dirxEntryUUID value in DirX Audit History Database.

- uuidfiltersize *size*
Specifies the size of searchFilter used, when retrieving entries from LDAP by their dirxEntryUUID values. A value of 100 means, that the searchFilter will contain 100 dirxEntryUUID values, and up to 100 entries might be returned in a single batch. Increasing the value improves performance, however, if there are errors on the LDAP server, try selecting a lower value. Default value is 100.

1.3.7.1. Fill Missing dirxEntryUUID Values of History Entries Examples

In this example, the tool looks for entries with missing dirxEntryUUID values in the DirX Audit History Database (either NULL or an empty string). For every entry with a missing value, the tool searches LDAP for a matching entry and if possible, uses its dirxEntryUUID value. If no matching entry is found, a new unique value is generated with a special prefix in the DirX Audit History Database (No modifications are made in LDAP). The connection data for the LDAP are stored in the ldap.properties file. The connection data for the tenant are stored in the configuration file of the tenant with ID **fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0**.

```
dxthistdbtool filluuid -ldapconfig ldap.properties -tenantid fb7b2fc2-a7e2-
4d9b-aea9-c002426d74c0
```

1.3.8. Make History Entries Unique

To ensure unique history entries in DirX Audit History Database (with unique and non-missing `dirxEntryUUID` values), execute the following command:

dxthistdbtool makeunique *common_options ldap_parameters*
[**-purge**]
[**-csvfile** *csvFile*]
[**-uuidfiltersize** *size*]

where

common_options
Specifies the common options. See the section "Common Options" for details.

ldap_parameters
Specifies LDAP connection parameters. See the section "LDAP Connection Parameters" for details.

-purge
Resolve exact duplicates (same on all attributes) by purging all extra entries. For example in case of three exact duplicates, two will be purged from the DirX Audit History Database, and only one will remain (the one with the smallest ENTRY_ID will usually remain).

-csvfile *csvFile*
Output digests (summaries) of duplicate entries found in the DirX Audit History Database. This parameter can be used in combination with the **-simulate** option to check what entries would be modified without actually making changes in the DirX Audit History Database.

- uuidfiltersize *size*
Specifies the size of searchFilter used, when retrieving entries from LDAP by their dirxEntryUUID values. A value of 100 means, that the searchFilter will contain 100 dirxEntryUUID values, and up to 100 entries might be returned in a single batch. Increasing the value improves performance, however, if there are errors on the LDAP server, try selecting a lower value. Default value is 100.

1.3.8.1. Make History Entries Unique Examples

In this example, the tool finds entries with duplicate dirxEntryUUID values in the DirX Audit History Database and generates new unique values. Entries with missing dirxEntryUUID values are resolved by internally calling the "filluuid" command. This is the default behavior. The connection data for the LDAP are stored in the ldap.properties file. The connection data for the tenant are stored in the configuration file of the tenant with ID **fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0**.

```
dxthistdbtool makeunique -ldapconfig ldap.properties -tenantid fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0
```

In this example, the tool first finds entries that are exact duplicates of each other (the same on every attribute), and purges all extra duplicate entries from the DirX Audit History

Database for the only configured tenant (if multiple tenants are configured you must specify the tenant ID). After this operation, the tool continues with the default behavior.

```
dxthistdbtool makeunique -ldapconfig ldap.properties -purge
```

In this example, the tool performs the operations described in the first example (the default behavior) and also outputs digests (summaries) of all duplicate entries found in the DirX Audit History Database to the specified CSV file for the only configured tenant (if multiple tenants are configured you must specify the tenant ID).

```
dxthistdbtool makeunique -ldapconfig ldap.properties -csvfile duplicates.csv
```

In this example, the tool makes no modifications to the DirX Audit History Database nor LDAP. It only outputs digests of all found duplicate entries into the specified CSV file. The connection data for the LDAP are stored in the ldap.properties file. The connection data for the tenant are stored in the configuration file of the tenant with ID **fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0**.

```
dxthistdbtool makeunique -ldapconfig ldap.properties -simulate -csvfile
duplicates.csv -tenantid fb7b2fc2-a7e2-4d9b-aea9-c002426d74c0
```

1.3.9. Import LDIF into DirX Audit History Database

To import LDAP entries from an LDIF file into the DirX Audit History Database, execute the following command:

dxthistdbtool importldif *common_options files*

where

common_options
Specifies the common options. See the section “Common Options” for details.

files
List of LDIF files to be imported into the DirX Audit History Database. Separate the filenames with a SPACE character. You can also specify directory names and use wildcard characters like *****. Each LDIF file must contain a single entry type. The entry type is extracted from the filename. For example: **account.ldif** imports entries of entry type **Account**. A prefix delimited by an underscore (_) can also be used. For example: **test_account.ldif** also imports entries of entry type **Account**.

1.3.9.1. Import LDIF into DirX Audit History Database Examples

In this example, the tool imports LDAP entries of entry type **User** from LDIF **user.ldif** and **Account** from LDIF **account.ldif** into the DirX Audit History Database. No changes to LDAP are made.

```
dxthistdbtool importldif user.ldif account.ldif
```

In this example, the tool imports all files inside the directory **/ldifs** with the extension **.ldif** into the DirX Audit History Database. The individual LDIF file names are used as entry types. No changes to LDAP are made.

```
dxthistdbtool importldif ./ldifs/*.ldif
```

1.4. Populate Fact Tables

The tool **db_fact_population** populates fact tables in the DirX Audit Data or History Database.
In day-to-day operations, the fact tables are filled regularly by the fact population component hosted on the DirX Audit Server. It calculates the facts for the last *n* days. The number of days and the schedule are configurable. See the chapter "Managing Fact and Dimension Tables" in the *DirX Audit Administration Guide* for more information. The fact population has two parts - Java based population and script-based population.

There are cases where periodic fact population is not enough; for example:

- After migrating to a new DirX Audit version, when you want to create facts in the new configuration for the audit messages that have already been written beforehand.
- You have exported old audit messages to files, re-imported them and want to have facts for them.

For this purpose, DirX Audit provides the **db_fact_population** command-line tool. It accepts a start and an end date and calculates the facts for this time range.

Note that the tool creates the fact and dimension tables according to the configuration when necessary. It also adds fact and dimension columns to existing tables when missing. But it doesn't delete existing columns from tables when they are removed from the configuration.

To populate fact tables in the DirX Audit Database, execute the following command:

dxtPopulateFacts *common_options*
[**-disableData**]
[**-disableHistory**]
[**-factTablesData** factTable [,factTable]...]
[**-factTablesHistory** factTable [,factTable]...]
[**-endDate** *yyyy.mm.dd*]
[**-startDate** *yyyy.mm.dd*]

where

common_options
Specifies the common options. See the section "Common Options" for details.

-disableData
Prevents the fact population operation for the DirX Audit Data Database from being initiated.

-disableHistory
Prevents the fact population operation for the DirX Audit History Database from being initiated.

-factTablesData *factTable*[,*factTable*]
Represents a list of DirX Audit Data Database fact tables to be populated. Separate fact table names with a comma and without spaces.

-factTablesHistory *factTable*[,*factTable*]
Represents a list of DirX Audit History Database fact tables to be populated. Separate fact table names with a comma and without spaces.

-**endDate**
Specifies the end of the time range for which the facts are to be populated using Java based population, including the end day. If it is missing, facts are calculated until the latest audit message.

-**startDate**
Specifies the beginning of the time range for which the facts are to be populated using Java based population. If it is missing, facts are calculated starting with the oldest audit message.

If both **-disableHistory** and **-disableData** options are used, no operation is performed.

1.4.1. Fact Table Population Examples

If you want to generate facts for all the audit messages in the tenant database (using Java based population; omit the start and end date as shown in the following command. Please make sure that you use the “simple plain hyphen” **-** in the command line.

```
dxtPopulateFacts -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
```

This example assumes that DirX Audit is installed and configured for multi-tenancy and you want to start fact population for a specific tenant. The connection data for the DirX Audit Data Database, DirX Audit History Database and localization of the configuration files for the fact and dimension tables are stored in the tenant configuration.

The next example calculates all the facts until the end of year 2011 (for Java-based population). The connection data and path to the fact configuration files are taken from the configuration file of the only configured tenant (stored configuration) and it starts without prompting the user to confirm the DirX Audit Data and History Databases to be updated (because the **-silent** option is used; see the section “Common Options” for details). If multiple tenants are configured you must specify the tenant ID.

```
dxtPopulateFacts -endDate 2011.12.31 -silent
```

The next example calculates the facts from January 1st until the end of March 2015 (for Java-based population). The connection data for the databases and localization of the configuration files are taken from the tenant configuration. Only the DirX Audit History Database is populated and the user is prompted to confirm the database to be updated (because the **-silent** option is not used; see the section "Common Options" for details):

```
dxtPopulateFacts -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99 -startDate
2015.01.01 -endDate 2015.03.31 -disableData
```

The next example calculates the facts for all the audit messages only in the DirX Audit Data Database. The connection data for the databases and localization of the configuration files are taken from the tenant configuration.

```
dxtPopulateFacts -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-disableHistory
```

The next example calculates the facts in the DirX Audit Data Database for specified fact tables (FCT_EVENTS, FCT_USERS) and their required dimensions. The DirX Audit History Database is not populated. The connection data for the databases and localization of the configuration files are taken from the tenant configuration. Note that if calculating The DirX Audit History Database is not disabled, all its fact will be calculated.

```
dxtPopulateFacts -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-factTablesData FCT_EVENTS,FCT_USERS -disableHistory
```

The next example calculates the facts in the DirX Audit History Database for specified fact tables (FCT_HST_ENTRIES, FCT_HST_USERS) and their required dimensions. The DirX Audit Data Database is not populated. The connection data for the databases and localization of the configuration files are taken from the tenant configuration. Note that if calculating The DirX Audit Data Database is not disabled, all its fact will be calculated.

```
dxtPopulateFacts -tenantid 4f753e1d-d0de-4aef-bb22-caace7342e99
-factTablesHistory FCT_HST_ENTRIES,FCT_HST_USERS -disableData
```

DirX Product Suite

The DirX product suite provides the basis for fully integrated identity and access management; it includes the following products, which can be ordered separately.

DirX Identity

DirX Identity provides a comprehensive, process-driven, customizable, cloud-enabled, scalable, and highly available identity management solution for businesses and organizations. It provides overarching, risk-based identity and access governance functionality seamlessly integrated with automated provisioning. Functionality includes lifecycle management for users and roles, cross-platform and rule-based real-time provisioning, web-based self-service functions for users, delegated administration, request workflows, access certification, password management, metadirectory as well as auditing and reporting functionality.

DirX Directory

DirX Directory provides a standards-compliant, high-performance, highly available, highly reliable, highly scalable, and secure LDAP and X.500 Directory Server and LDAP Proxy with very high linear scalability. DirX Directory can serve as an identity store for employees, customers, partners, subscribers, and other IoT entities. It can also serve as a provisioning, access management and metadirectory repository, to provide a single point of access to the information within disparate and heterogeneous directories available in an enterprise network or cloud environment for user management and provisioning.

DirX Access

DirX Access is a comprehensive, cloud-ready, scalable, and highly available access management solution providing policy- and risk-based authentication, authorization based on XACML and federation for Web applications and services. DirX Access delivers single sign-on, versatile authentication including FIDO, identity federation based on SAML, OAuth and OpenID Connect, just-in-time provisioning, entitlement management and policy enforcement for applications and services in the cloud or on-premises.

DirX Audit

DirX Audit provides auditors, security compliance officers and audit administrators with analytical insight and transparency for identity and access. Based on historical identity data and recorded events from the identity and access management processes, DirX Audit allows answering the “what, when, where, who and why” questions of user access and entitlements. DirX Audit features historical views and reports on identity data, a graphical dashboard with drill-down into individual events, an analysis view for filtering, evaluating, correlating, and reviewing of identity-related events and job management for report generation.

For more information: support.dirx.solutions/about

Legal remarks

On the account of certain regional limitations of sales rights and service availability, we cannot guarantee that all products included in this document are available through the Eviden sales organization worldwide. Availability and packaging may vary by country and is subject to change without prior notice. Some/All of the features and products described herein may not be available locally. The information in this document contains general technical descriptions of specifications and options as well as standard and optional features which do not always have to be present in individual cases. Eviden reserves the right to modify the design, packaging, specifications and options described herein without prior notice. Please contact your local Eviden sales representative for the most current information. Note: Any technical data contained in this document may vary within defined tolerances. Original images always lose a certain amount of detail when reproduced.