Data Loader Guide

Version 46.0, Summer '19

@salesforcedocs

Last updated: August 15, 2019

© Copyright 2000–2019 salesforce.com, inc. All rights reserved. Salesforce is a registered trademark of salesforce.com, inc., as are other names and marks. Other marks appearing herein may be trademarks of their respective owners.

CONTENTS

Chapter 1: Data Loader 1

Chapter 2: When to Use Data Loader 2

Chapter 3: Installing Data Loader 3

Install Data Loader on macOS 4
Install Data Loader on Windows 5
Considerations for Installing Data Loader 6

Chapter 4: Configure Data Loader 8

Data Loader Behavior with Bulk API Enabled 13
Configure the Data Loader to Use the Bulk API 13

Chapter 5: Using Data Loader 14

Data Types Supported by Data Loader 15
Export Data 16
Define Data Loader Field Mappings 18
Insert, Update, or Delete Data Using Data Loader 18
    Perform Mass Updates 20
    Perform Mass Deletes 20
Upload Attachments 20
Upload Content with the Data Loader 21
Review Data Loader Output Files 22
Data Import Dates 23
View the Data Loader Log File 23
Configure the Data Loader Log File 24

Chapter 6: Running in Batch Mode (Windows Only) 25

Installed Directories and Files 26
Encrypt from the Command Line 26
Upgrade Your Batch Mode Interface 27
Run Batch File With Windows Command-Line Interface 27
Configure Batch Processes 28
Data Loader Process Configuration Parameters 29
Data Loader Command-Line Operations 37
Configure Database Access 38
    Spring Framework 39
    Data Access Objects 40
    SQL Configuration 40
Map Columns 42

Run Individual Batch Processes 44

Chapter 7: Command-Line Quick Start (Windows Only) 45

Data Loader Command Line Introduction 46
Prerequisites 46
Step One: Create the Encryption Key File 47
Step Two: Create the Encrypted Password 48
Step Three: Create the Field Mapping File 48
Step Four: Create the Configuration File 49
Step Five: Import the Data 51

Appendix A: Data Loader Third-Party Licenses 52

Index 53

CHAPTER 1 Data Loader

Data Loader is a client application for the bulk import or export of data. Use it to insert, update, delete, or export Salesforce records.

When importing data, Data Loader reads, extracts, and loads data from comma-separated values (CSV) files or from a database connection. When exporting data, it outputs CSV files.

Note: If commas aren't appropriate for your locale, use a tab or other delimiter. Specify your delimiter in Data Loader Settings (**Settings** | **Settings**).

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

You can use Data Loader in two different ways:

- User interface—When you use the user interface, you work interactively to specify the configuration parameters, CSV files used for import and export, and the field mappings that map the field names in your import file with the field names in Salesforce.
- Command line (Windows only)—When you use the command line, you specify the configuration, data sources, mappings, and actions in files. This enables you to set up Data Loader for automated processing.

Data Loader offers the following key features:

- An easy-to-use wizard interface for interactive use
- An alternate command-line interface for automated batch operations (Windows only)
- Support for large files with up to 5 million records
- Drag-and-drop field mapping
- Support for all objects, including custom objects
- Can be used to process data in both Salesforce and Database.com
- Detailed success and error log files in CSV format
- A built-in CSV file viewer
- Support for Windows and Mac

To get started, see the following topics:

- When to Use Data Loader
- Considerations for Installing Data Loader

Note: In previous versions, Data Loader has been known as "AppExchange Data Loader" and "Sforce Data Loader."

CHAPTER 2 When to Use Data Loader

Data Loader complements the web-based import wizards that are accessible from the Setup menu in the online application. Refer to the following guidelines to determine which method best suits your business needs:

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Use Data Loader when:

- You need to load 50,000 to 5,000,000 records. Data Loader is supported for loads of up to 5 million records. If you need to load more than 5 million records, we recommend you work with a Salesforce partner or visit the *AppExchange* for a suitable partner product.
- You need to load into an object that is not yet supported by the import wizards.
- You want to schedule regular data loads, such as nightly imports.
- You want to export your data for backup purposes.

Use the import wizards when:

- You are loading less than 50,000 records.
- The object you need to import is supported by import wizards. To see what import wizards are available and thus what objects they support, from Setup, enter `Data Management` in the `Quick Find` box, then select **Data Management**.
- You want to prevent duplicates by uploading records according to account name and site, contact email address, or lead email address.

For more information about the import wizards, see Import Data Into Salesforce.

CHAPTER 3 Installing Data Loader

In this chapter ...

- Install Data Loader on macOS
- Install Data Loader on Windows
- Considerations for Installing Data Loader

You can use Data Loader to insert, update, delete, and export Salesforce records.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Install Data Loader on macOS

You can use Data Loader to insert, update, delete, and export Salesforce records. To use it, you install it on your macOS computer.

1. Install Zulu OpenJDK version 11for macOS using the .DMG file.
2. From Setup, download the Data Loader installation file.
3. Right-click the .zip file and select **Extract All**.
4. In the Data Loader folder, double-click the **installer.command** file. If you can't open the file because of an unidentified developer message, press the Control key while clicking the **installer.command** file, and select **open** from the menu.
5. Specify where to install Data Loader.
6. Specify whether to add a Data Loader launch icon on your desktop or Applications folder. Data Loader completes the installation.

To run Data Loader, use the Data Loader desktop icon, Applications folder, or dataloader.command in your installation folder. If Zulu OpenJDK is not found, a message prompts you to download and install it.

Note: To continue using your settings from an earlier Data Loader version in version 45 and later, migrate your config and log files after installation. Search for `config.properties` and `log-config.xml`, and copy or move the files to your current Data Loader's `/config` folder.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

USER PERMISSIONS

To access the page to download Data Loader:

- Modify All Data

To use Data Loader:

- API Enabled

  AND

  The appropriate user permission for the operation you are doing, for example, Create on accounts to insert new accounts

  AND

  Bulk API Hard Delete (only if you configure Data Loader to use Bulk API to hard-delete records)

Install Data Loader on Windows

You can use Data Loader to insert, update, delete, and export Salesforce records. To use it, you install it on your Windows computer.

1. Install Zulu OpenJDK version 11 for Windows using the .MSI file.
2. From Setup, download the Data Loader installation file.
3. Right-click the .zip file and select **Extract All**.
4. In the Data Loader folder, double-click the **install.bat** file. If you get an unknown publisher error message, you can ignore it and run the file.
5. Specify where to install Data Loader, and select whether to overwrite an existing Data Loader installation.
6. Specify whether to create a Data Loader launch icon on your desktop or a start menu shortcut. Data Loader completes the installation.

To run Data Loader, use the Data Loader desktop icon, start menu entry, or the dataloader.bat file in your installation folder. If Zulu OpenJDK is not found, a message prompts you to download and install it.

Note: To continue using your settings from an earlier Data Loader version in version 45 and later, migrate your config and log files after installation. Search for `config.properties` and `log-config.xml`, and copy or move the files to your current Data Loader's `/config` folder.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

USER PERMISSIONS

To access the page to download Data Loader:

- Modify All Data

To use Data Loader:

- API Enabled

  AND

  The appropriate user permission for the operation you are doing, for example, Create on accounts to insert new accounts

  AND

  Bulk API Hard Delete (only if you configure Data Loader to use Bulk API to hard-delete records)

Considerations for Installing Data Loader

Before you download and install Data Loader, understand the system requirements, installation considerations, and login considerations.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

USER PERMISSIONS

To access the page to download Data Loader:
- Modify All Data

To use Data Loader:
- API Enabled

  AND

  The appropriate user permission for the operation you are doing, for example, Create on accounts to insert new accounts

  AND

  Bulk API Hard Delete (only if you configure Data Loader to use Bulk API to hard-delete records)

System Requirements for Windows

Data Loader is signed for Windows. To use Data Loader for Windows, you need:

- 64-bit Microsoft® Windows® 7, Windows 8, or Windows 10
- 120 MB of free disk space
- 256 MB of available memory
- Zulu OpenJDK version 11

Note: Salesforce no longer bundles Java with the Data Loader for Windows installer. Download and install Java on your Windows computer.

We recommend that you set the `JAVA_HOME` environment variable to the directory where Zulu OpenJDK is installed. Doing so ensures that you can run Data Loader in batch mode from the command line.

System Requirements for macOS

To use Data Loader for macOS, you need:

- 64-bit macOS El Capitan
- 120 MB of free disk space
- 256 MB of available memory
- Zulu OpenJDK version 11

Installation Considerations

Over time, several versions of the Data Loader client application have been available for download. Some earlier versions were called "AppExchange Data Loader" or "Sforce Data Loader." You can run different versions at the same time on one computer. However, do not install more than one copy of the same version. If you have installed the latest version and want to install it again, first remove the version on your computer.

The latest version is always available in Salesforce. From Setup, enter *Data Loader* in the Quick Find box, then select **Data Loader**.

Note: Install Zulu OpenJDK version 11 before installing Data Loader.

Tip: If you experience login issues in the command-line interface after upgrading Data Loader, try encrypting your password again to solve the problem. For information, see Encrypt from the Command Line on page 26.

Note: The Data Loader command-line interface is supported for Windows only.

To change the source code, download the open-source version of Data Loader from *https://github.com/forcedotcom/dataloader*.

Login Considerations

- If your organization restricts IP addresses, logins from untrusted IPs are blocked until they're activated. Salesforce automatically sends you an activation email that you can use to log in. The email contains a security token that you must add to the end of your password. For example, if your password is *mypassword*, and your security token is *XXXXXXXXXX*, you must enter *mypasswordXXXXXXXXXX* to log in.
- Data Loader version 36.0 and later supports Web Server OAuth Authentication. See OAuth Authentication for more information.
- Data Loader version 36.0 and later supports Salesforce Communities. Communities users always log in with the OAuth option in Data Loader. To enable OAuth for Communities, the user modifies the `config.properties` file as follows.
  - Change the portion in bold in the following line to the login URL of the community. Don't add a forward slash (/) to the end of the line.

    ```
    sfdc.oauth.Production.server=https\://login.salesforce.com
    ```

    For example:

    ```
    sfdc.oauth.Production.server=
    https\://johnsmith-developer-edition.yourInstance.force.com/test
    ```

  - Change the portion in bold in the following line to the hostname of the community.

    ```
    sfdc.oauth.Production.redirecturi=https\://login.salesforce.com/services/oauth2/success
    ```

    For example:

    ```
    sfdc.oauth.Production.redirecturi=
    https\:/johnsmith-developer-edition.yourInstance.force.com/services/oauth2/success
    ```

  The `config.properties` file is in the `configs` default configuration directory, which is installed in these locations.

  - macOS: `/Users/{userName}/dataloader/version/configs`
  - Windows: `C:\Users\{userName}\dataloader\version\configs`

CHAPTER 4 Configure Data Loader

In this chapter ...

- Data Loader Behavior with Bulk API Enabled
- Configure the Data Loader to Use the Bulk API

Use the Settings menu to change the Data Loader default operation settings.

1. Open the Data Loader.
2. Select **Settings** > **Settings**.
3. Edit the fields as needed.

Field	Description
`Batch size`	In a single insert, update, upsert, or delete operation, records moving to or from Salesforce are processed in increments of this size. The maximum is 200 records. We recommend a value between 50 and 100. The maximum value is 10,000 if the `Use Bulk API` option is selected.
`Insert null values`	Select this option to insert blank mapped values as `null` values during data operations. When you are updating records, this option instructs Data Loader to overwrite existing data in mapped fields. This option is not available if the `Use Bulk API` option is selected. Empty field values are ignored when you update records using the Bulk API. To set a field value to `null` when the `Use Bulk API` option is selected, use a field value of `#N/A`.
`Assignment rule`	Specify the ID of the assignment rule to use for inserts, updates, and upserts. This option applies to inserts,

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Field	Description
	updates, and upserts on cases and leads. With the original territory management feature, it also applies to updates on accounts if your organization has territory assignment rules on accounts. The assignment rule overrides `Owner` values in your CSV file.
`Server host`	Enter the URL of the Salesforce server with which you want to communicate. For example, if you are loading data into a sandbox, change the URL to `https://test.salesforce.com`.
`Reset URL on Login`	By default, Salesforce resets the URL after login to the one specified in `Server host`. To turn off this automatic reset, disable this option.
`Compression`	Compression enhances the performance of Data Loader and is turned on by default. You might want to disable compression when debugging the underlying SOAP messages. To turn off compression, enable this option.
`Timeout`	Specify how many seconds Data Loader waits to receive a response back from the server before returning an error for the request.
`Query request size`	In a single export or query operation, records are returned from Salesforce in increments of this size. Larger values can improve performance but use more memory on the client. The default is 500; the minimum is 200, and the maximum is 2,000. There is no guarantee that the requested batch size requested is the actual batch size; changes are sometimes made to maximize performance.
`Generate status files for exports`	Select this option to generate success and error files when exporting data.
`Read all CSVs with UTF-8 encoding`	Select this option to force files to open in UTF-8 encoding, even if they were saved in a different format.
`Write all CSVs with UTF-8 encoding`	Select this option to force files to be written in UTF-8 encoding.
`Use European date format`	Select this option to support the date formats `dd/MM/yyyy` and `dd/MM/yyyy HH:mm:ss`.

Field	Description
`Allow field truncation`	Select this option to truncate data in the following types of fields when loading that data into Salesforce: Email, Multi-select Picklist, Phone, Picklist, Text, and Text (Encrypted). In Data Loader versions 14.0 and earlier, Data Loader truncates values for fields of those types if they are too large. In Data Loader version 15.0 and later, the load operation fails if a value is specified that is too large. Selecting this option allows you to specify that the previous behavior, truncation, be used instead of the new behavior in Data Loader versions 15.0 and later. This option is selected by default and has no effect in versions 14.0 and earlier. This option is not available if the `Use Bulk API` option is selected. In that case, the load operation fails for the row if a value is specified that is too large for the field.
`Allow comma as a CSV delimiter`	Select this option if your CSV file uses commas to delimit records.
`Allow tab as a CSV delimiter`	Select this option if your CSV file uses tab characters to delimit records.
`Allow other characters as CSV delimiters`	Select this option if your CSV file uses a character other than a comma or tab to delimit records.
`Other delimiters (enter multiple values with no separator; for example, !+?)`	The characters in this field are used only if the **Allow other characters as CSV delimiters** option is selected. For example, if you use the | (pipe) character to delimit data records, enter that character in this field.
`Use Bulk API`	Select this option to use Bulk API to insert, update, upsert, delete, and hard-delete records. Bulk API is optimized to load or delete many records asynchronously. It's faster than the default SOAP-based API due to parallel processing and fewer network round-trips. Warning: You can hard delete records when you configure Data Loader to `Use Bulk API`. Keep in mind that hard deleted records are immediately deleted and can't be recovered from the Recycle Bin.

Field	Description
`Enable serial mode for Bulk API`	To use serial processing instead of parallel processing for Bulk API, select this option. Processing in parallel can cause database contention. When contention is severe, the load can fail. Serial mode processes batches one at a time, however it can increase the processing time for a load. This option is only available if the `Use Bulk API` option is selected.
`Upload Bulk API Batch as Zip File`	Select this option to use Bulk API to upload zip files containing binary attachments, such as Attachment records or Salesforce CRM Content. This option is only available if the `Use Bulk API` option is selected.
`Time Zone`	Select this option to specify a default time zone. If a date value does not include a time zone, this value is used. • If no value is specified, the time zone of the computer where Data Loader is installed is used. • If an incorrect value is entered, GMT is used as the time zone and this fact is noted in the Data Loader log. Valid values are any time zone identifier which can be passed to the Java `getTimeZone(java.lang.String)` method. The value can be a full name such as `America/Los_Angeles`, or a custom ID such as `GMT-8:00`.
`Proxy host`	The host name of the proxy server, if applicable.
`Proxy port`	The proxy server port.
`Proxy username`	The username for proxy server authentication.
`Proxy password`	The password for proxy server authentication.
`Proxy NTLM domain`	The name of the Windows domain used for NTLM authentication.
`Start at row`	If your last operation failed, you can use this setting to begin where the last successful operation finished.

4. Click **OK** to save your settings.

Data Loader Behavior with Bulk API Enabled

Enabling the Bulk API in Data Loader allows you to load or delete a large number of records faster than using the default SOAP-based API. However, there are some differences in behavior in Data Loader when you enable the Bulk API. One important difference is that it allows you to execute a hard delete if you have the permission and license. See Configure Data Loader on page 8.

The following settings are not available on the **Settings** > **Settings** page in Data Loader when the `Use Bulk API` option is selected:

`Insert null values`
This option enables Data Loader to insert blank mapped values as `null` values during data operations when the Bulk API is disabled. Empty field values are ignored when you update records using the Bulk API. To set a field value to `null` when the `Use Bulk API` option is selected, use a field value of `#N/A`.

`Allow field truncation`
This option directs Data Loader to truncate data for certain field types when the Bulk API is disabled. A load operation fails for the row if a value is specified that is too large for the field when the `Use Bulk API` option is selected.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Configure the Data Loader to Use the Bulk API

The Bulk API is optimized to load or delete a large number of records asynchronously. It is faster than the SOAP-based API due to parallel processing and fewer network round-trips. By default, Data Loader uses the SOAP-based API to process records.

To configure Data Loader to use the Bulk API for inserting, updating, upserting, deleting, and hard deleting records:

1. Open the Data Loader.
2. Choose **Settings** > **Settings**.
3. Select the `Use Bulk API` option.
4. Click **OK**.

Note:

- You can also select the `Enable serial mode for Bulk API` option. Processing in parallel can cause database contention. When contention is severe, the load can fail. Serial mode processes batches one at a time, however it can increase the processing time for a load.
- **Caution:** You can hard delete records when you configure Data Loader to `Use Bulk API`. Keep in mind that hard deleted records are immediately deleted and can't be recovered from the Recycle Bin.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

CHAPTER 5 Using Data Loader

In this chapter ...

- Data Types Supported by Data Loader
- Export Data
- Define Data Loader Field Mappings
- Insert, Update, or Delete Data Using Data Loader
- Upload Attachments
- Upload Content with the Data Loader
- Review Data Loader Output Files
- Data Import Dates
- View the Data Loader Log File
- Configure the Data Loader Log File

Using Data Loader, you can perform various operations which include exporting data, defining field mappings, inserting, updating, and deleting data, performing mass updates and mass deletes, uploading attachments and content, and reviewing output files.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Data Types Supported by Data Loader

Data Loader supports the following data types:

Base64

String path to file (converts the file to a base64–encoded array). Base64 fields are only used to insert or update attachments and Salesforce CRM Content. For more information, see Upload Attachments on page 20 and Upload Content with the Data Loader on page 21.

Boolean

- True values (case insensitive) = `yes`, `y`, `true`, `on`, `1`
- False values (case insensitive) = `no`, `n`, `false`, `off`, `0`

Date Formats

We recommend you specify dates in the format *`yyyy-MM-ddTHH:mm:ss.SSS+/-HHmm`*:

- `yyyy` is the four-digit year
- `MM` is the two-digit month (01-12)
- `dd` is the two-digit day (01-31)
- `HH` is the two-digit hour (00-23)
- `mm` is the two-digit minute (00-59)
- `ss` is the two-digit seconds (00-59)
- `SSS` is the three-digit milliseconds (000-999)
- `+/-HHmm` is the Zulu (UTC) time zone offset

The following date formats are also supported:

- `yyyy-MM-dd'T'HH:mm:ss.SSS'Z'`
- `yyyy-MM-dd'T'HH:mm:ss.SSS Pacific Standard Time`
- `yyyy-MM-dd'T'HH:mm:ss.SSSPacific Standard Time`
- `yyyy-MM-dd'T'HH:mm:ss.SSS PST`
- `yyyy-MM-dd'T'HH:mm:ss.SSSPST`
- `yyyy-MM-dd'T'HH:mm:ss.SSS GMT-08:00`
- `yyyy-MM-dd'T'HH:mm:ss.SSSGMT-08:00`
- `yyyy-MM-dd'T'HH:mm:ss.SSS -800`
- `yyyy-MM-dd'T'HH:mm:ss.SSS-800`
- `yyyy-MM-dd'T'HH:mm:ss`
- `yyyy-MM-dd HH:mm:ss`
- `yyyyMMdd'T'HH:mm:ss`
- `yyyy-MM-dd`
- `MM/dd/yyyy HH:mm:ss`
- `MM/dd/yyyy`
- `yyyyMMdd`

Note the following tips for date formats:

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

- To enable date formats that begin with the day rather than the month, select the `Use European date format` box in the Settings dialog. European date formats are `dd/MM/yyyy` and `dd/MM/yyyy HH:mm:ss`.
- If your computer's locale is east of Greenwich Mean Time (GMT), we recommend that you change your computer setting to GMT in order to avoid date adjustments when inserting or updating records.
- Only dates within a certain range are valid. The earliest valid date is 1700-01-01T00:00:00Z GMT, or just after midnight on January 1, 1700. The latest valid date is 4000-12-31T00:00:00Z GMT, or just after midnight on December 31, 4000. These values are offset by your time zone. For example, in the Pacific time zone, the earliest valid date is 1699-12-31T16:00:00, or 4:00 PM on December 31, 1699.

Double
Standard double string

ID
A Salesforce ID is a case-sensitive 15-character or case–insensitive 18-character alphanumeric string that uniquely identifies a particular record.

Tip: To ensure data quality, make sure that all Salesforce IDs you enter in Data Loader are in the correct case.

Integer
Standard integer string

String
All valid XML strings; invalid XML characters are removed.

Export Data

You can use the Data Loader export wizard to extract data from a Salesforce object.

1. Open the Data Loader.
2. Click **Export**. If you want to also export archived activity records and soft-deleted records, click **Export All** instead.
3. Enter your Salesforce username and password, and click **Log in**.
4. When you're logged in, click **Next**. (You are not asked to log in again until you log out or close the program.)

   If your organization restricts IP addresses, logins from untrusted IPs are blocked until they're activated. Salesforce automatically sends you an activation email that you can use to log in. The email contains a security token that you must add to the end of your password. For example, if your password is *`mypassword`*, and your security token is *`XXXXXXXXXX`*, you must enter *`mypasswordXXXXXXXXXX`* to log in.
5. Choose an object. For example, select the Account object. If your object name isn't listed, select **Show all objects** to see all the objects that you can access. The objects are listed by localized label name, with the developer name in parentheses. For object descriptions, see the *SOAP API Developer Guide*.
6. Select the CSV file to export the data to. You can choose an existing file or create a file.

   If you select an existing file, the export replaces its contents. To confirm the action, click **Yes**, or choose another file by clicking **No**.
7. Click **Next**.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

USER PERMISSIONS

To export records:
- Read on the records

To export all records:
- Read on the records

8. Create a SOQL query for the data export. For example, select **Id** and **Name** in the query fields, and click **Finish**. As you follow the next steps, the CSV viewer displays all the Account names and their IDs. SOQL is the Salesforce Object Query Language. Similar to the SELECT command in SQL, with SOQL, you can specify the source object, a list of fields to retrieve, and conditions for selecting rows in the source object.

   a. Choose the fields you want to export.

   b. Optionally, select conditions to filter your dataset. If you do not select any conditions, all the data to which you have read access is returned.

   c. Review the generated query and edit if necessary.

   Tip: You can use a SOQL relationship query to include fields from a related object. For example:

   ```
   Select Name, Pricebook2Id, Pricebook2.Name, Product2Id, Product2.ProductCode FROM
   PricebookEntry WHERE IsActive = true
   ```

   Or:

   ```
   Select Id, LastName, Account.Name FROM Contact
   ```

   When using relationship queries in the Data Loader, the fully specified field names are case-sensitive. For example, using `ACCOUNT.NAME` instead of `Account.Name` does not work.

   Data Loader doesn't support nested queries or querying child objects. For example, queries similar to the following return an error:

   ```
   SELECT Amount, Id, Name, (SELECT Quantity, ListPrice,
   PriceBookEntry.UnitPrice, PricebookEntry.Name,
   PricebookEntry.product2.Family FROM OpportunityLineItems)
   FROM Opportunity
   ```

   Also, Data Loader doesn't support queries that use polymorphic relationships. For example, the following query results in an error:

   ```
   SELECT Id, Owner.Name, Owner.Type, Owner.Id, Subject FROM Case
   ```

   For more information on SOQL, see the *SOQL and SOSL Reference*.

9. Click **Finish**, then click **Yes** to confirm.

   A progress information window reports the status of the operation. After the operation completes, a confirmation window summarizes your results.

10. To view the CSV file. click **View Extraction**, or to close, click **OK**. For more details, see Review Data Loader Output Files on page 22.

Note:

- Data Loader currently does not support exporting attachments. As a workaround, use the weekly export feature in the online application to export attachments.
- If you select compound fields for export in the Data Loader, they cause error messages. To export values, use individual field components.

Define Data Loader Field Mappings

When you insert, delete, or update files, use the Mapping Dialog window to associate Salesforce fields with the columns of your CSV file. For more information, see Insert, Update, or Delete Data Using Data Loader on page 18.

1. To automatically match fields with columns, click **Auto-Match Fields to Columns**. The Data Loader populates the list at the bottom of the window based on the similarity of field and column names. For a delete operation, automatic matching works only on the ID field.
2. To manually match fields with columns, click and drag fields from the list of Salesforce fields at the top to the list of CSV column header names at the bottom. For example, if you are inserting new Account records where your CSV file contains the names of new accounts, click and drag the `Name` field to the right of the `NAME` column header field.
3. Optionally, click **Save Mapping** to save this mapping for future use. Specify a name for the SDL mapping file.

   If you select an existing file, the export replaces its contents. To confirm the action, click **Yes**, or choose another file by clicking **No**.
4. Click **OK** to use your mapping for the current operation.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Insert, Update, or Delete Data Using Data Loader

USER PERMISSIONS

To insert records:	Create on the record
To update records:	Edit on the record
To upsert records:	Create or Edit on the record
To delete records:	Delete on the record
To hard delete records:	Delete on the record
To mass delete records:	Modify All Data

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Use the Data Loader wizards to add, modify, or delete records. The upsert wizard combines inserting and updating a record. If a record in your file matches an existing record, the existing record is updated with the values in your file. If no match is found, a new record is created. When you hard-delete records, the deleted records are not stored in the Recycle Bin and are eligible for deletion. For more information, see Configure Data Loader.

1. Open the Data Loader.
2. Click **Insert**, **Update**, **Upsert**, **Delete**, or **Hard Delete**. These commands are also listed in the File menu.
3. Enter your Salesforce username and password. To log in, click **Log in**. When you are logged in, click **Next**. (Until you log out or close the program, you are not asked to log in again.)

   If your organization restricts IP addresses, logins from untrusted IPs are blocked until they're activated. Salesforce automatically sends you an activation email that you can use to log in. The email contains a security token that you must add to the end of your password. For example, if your password is *`mypassword`*, and your security token is *`XXXXXXXXXX`*, you must enter *`mypasswordXXXXXXXXXX`* to log in.

4. Choose an object. For example, if you are inserting Account records, select **Account**. If your object name does not display in the default list, select **Show all objects** to see a complete list of the objects that you can access. The objects are listed by localized label name, with the developer name noted in parentheses. For object descriptions, see the *Object Reference for Salesforce and Lightning Platform*.

   Note: Data Loader deletes records based on the IDs in the CSV file, not the object selected.

5. To select your CSV file, click **Browse**. For example, if you are inserting Account records, you could specify a CSV file called `insertaccounts.csv` containing a Name column for the names of the new accounts.
6. Click **Next**. After the object and CSV file are initialized, click **OK**.
7. If you are performing an upsert, your CSV file must contain a column of ID values for matching against existing records. The column is either an external ID (a custom field with the External ID attribute) or ID (the Salesforce record ID).
   a. From the dropdown list, select which field to use for matching. If the object has no external ID fields, ID is used. Click **Next** to continue.
   b. If your file includes the external IDs of an object that has a relationship to your chosen object, enable that external ID for record matching by selecting its name from the dropdown list. If you make no selection, you can use the related object's ID field for matching by mapping it in the next step. Click **Next** to continue.
8. Define how the columns in your CSV file map to Salesforce fields. To select an existing field mapping, click **Choose an Existing Map**. To create or modify a map, click **Create or Edit a Map**. For more information, see Define Data Loader Field Mappings on page 18. Click **Next**.
9. For each operation, the Data Loader generates two unique CSV log files. One file name starts with "success," and the other starts with "error." Click **Browse** to specify a directory for these files.
10. To complete the operation, click **Finish**, and then click **Yes** to confirm. As the operation proceeds, a progress information window reports the status of the data movement.
11. To view your success or error files, click **View Successes** or **View Errors**. To close the wizard, click **OK** . For more information, see Review Data Loader Output Files on page 22.

Tip:

- If you are updating or deleting large amounts of data, review Perform Mass Updates and Perform Mass Deletes for tips and best practices.
- There is a 5-minute limit to process 100 records when the Bulk API is enabled. If it takes longer than 10 minutes to process a file, the Bulk API places the remainder of the file back in the queue for later processing. If the Bulk API continues to exceed the 10-minute limit on subsequent attempts, the file is placed back in the queue and reprocessed up to 10 times before the operation is permanently marked as failed. Even if the processing fails, some records could have completed successfully, so check the results. If you get a timeout error when loading a file, split your file into smaller files and try again.

Perform Mass Updates

To update a large number of records at one time, we recommend the following steps:

1. Obtain your data by performing an export of the objects you wish to update, or by running a report. Make sure your report includes the record ID.
2. As a backup measure, save an extra copy of the generated CSV file.
3. Open your working file in a CSV editor such as Excel, and update your data.
4. Launch Data Loader and follow the update wizard. Note that matching is done according to record ID. See Insert, Update, or Delete Data Using Data Loader on page 18.
5. After the operation, review your success and error log files. See Review Data Loader Output Files on page 22.
6. If you made a mistake, use the backup file to update the records to their previous values.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Perform Mass Deletes

To delete a large number of records at one time using Data Loader, we recommend the following steps:

1. As a backup measure, export the records you wish to delete, being sure to select all fields. (See Export Data on page 16.) Save an extra copy of the generated CSV file.
2. Next, export the records you wish to delete, this time using only the record ID as the desired criterion.
3. Launch the Data Loader and follow the delete or hard delete wizard. Map only the ID column. See Insert, Update, or Delete Data Using Data Loader on page 18.
4. After the operation, review your success and error log files. See Review Data Loader Output Files on page 22.

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

USER PERMISSIONS

To mass delete records:
- Modify All Data

Upload Attachments

Use Data Loader to upload attachments to Salesforce.

Before uploading attachments, note the following:

- If you intend to upload with Bulk API, verify that `Upload Bulk API Batch as Zip File` on the **Settings** > **Settings** page is enabled.
- If you are migrating attachments from a source Salesforce org to a target org, begin by requesting a data export for the source org. On the Schedule Export page, select **Include Attachments** to include the `Attachment.csv` file in your export. You can use this CSV file to upload the attachments. For more information on the export service, see Export Backup Data from Salesforce.

1. Confirm that the CSV file you want to use for attachment importing contains these required columns. Each column represents a Salesforce field.
   - ParentId—Salesforce ID of the parent record
   - Name—Name of the attachment file, such as `myattachment.jpg`
   - Body—Absolute path to the attachment on your local drive

Make sure that the values in the Body column contain the full path of the attachments on your computer. For example, if an attachment named `myattachment.jpg` is the folder `C:\Export`, Body must specify `C:\Export\myattachment.jpg`. Your CSV file looks like this example:

```
ParentId,Name,Body
50030000000VDowAAG,attachment1.jpg,C:\Export\attachment1.gif
701300000000iNHAAY,attachment2.doc,C:\Export\files\attachment2.doc
50030000000VDowAAG,attachment1.jpg,C:\Export\attachment word document.doc
```

The CSV file can also include other optional Attachment fields, such as Description.

2. Proceed with an insert or upsert operation (see Insert, Update, or Delete Data Using Data Loader on page 18). For the select data objects step, select **Show all Salesforce objects** and the attachment object name in the list.

Upload Content with the Data Loader

You can use Data Loader to bulk upload documents and links into libraries in Salesforce CRM Content. Before uploading documents or links, note the following.

- If you intend to upload with Bulk API, verify that `Upload Bulk API Batch as Zip File` on the **Settings** > **Settings** page is enabled.
- When you upload a document from your local drive using Data Loader, specify the path in the `VersionData` and `PathOnClient` fields in the CSV file. `VersionData` identifies the location and extracts the format, and `PathOnClient` identifies the type of document being uploaded.
- When you upload a link using the Data Loader, specify the URL in `ContentUrl`. Don't use `PathOnClient` or `VersionData` to upload links.
- You can't export content using the Data Loader.
- If you're updating content that you've already uploaded:
  - Perform the Insert function.
  - Include a `ContentDocumentId` column with an 18-character ID. Salesforce uses this information to determine that you're updating content. When you map the `ContentDocumentId`, the updates are added to the content file. If you don't include the ContentDocumentId, the content is treated as new, and the content file isn't updated.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

1. Create a CSV file with the following fields.
   - `Title` - file name.
   - `Description` - (optional) file or link description.

     Note: If there are commas in the description, use double quotes around the text.

   - `VersionData` - complete file path on your local drive (for uploading documents only).

     Note: Files are converted to base64 encoding on upload. This action adds approximately 30% to the file size.

   - `PathOnClient` - complete file path on your local drive (for uploading documents only).
   - `ContentUrl` - URL (for uploading links only).
   - `OwnerId` - (optional) file owner, defaults to the user uploading the file.
   - `FirstPublishLocationId` - library ID.

- `RecordTypeId` - record type ID.

  Note: If you publish to a library that has restricted record types, specify `RecordTypeId`.

  To determine the `RecordTypeId` values for your organization using Data Loader, follow the steps in Exporting Data. The following is a sample SOQL query:

  ```
  Select Id, Name FROM RecordType WHERE SobjectType = 'ContentVersion'
  ```

  To determine the `RecordTypeId` values for your organization using the AJAX Toolkit:

  a. Log in to Salesforce.

  b. Enter this URL in your browser:
  `http://instanceName.salesforce.com/soap/ajax/46.0/debugshell.html`. Enter the *instanceName* for your organization. You can see the *instanceName* in the URL field of your browser after logging in to Salesforce.

  c. In the AJAX Toolkit Shell page, type:

  ```
  sforce.connection.describeSObject("ContentVersion")
  ```

  d. Press **Enter**.

  e. Click the arrows for `recordTypeInfos`.

  The `RecordTypeId` values for your organization are listed.

- `TagsCsv` - (optional) tag.

A sample CSV file is:

```
Title,Description,VersionData,PathOnClient,OwnerId,FirstPublishLocationId,RecordTypeId,TagsCsv
testfile,"This is a test file, use for bulk
upload",c:\files\testfile.pdf,c:\files\testfile.pdf,005000000000000,058700000004Cd0,012300000008o2sAQG,one
```

2. Upload the CSV file for the ContentVersion object (see Insert, Update, or Delete Data Using Data Loader on page 18). All documents and links are available in the specified library.

Review Data Loader Output Files

After an import or export, Data Loader generates two CSV output files that contain the results of the operation. One file name starts with "success," and the other starts with "error." You can use the Data Loader CSV file viewer to open the files.

1. Choose **View** > **View CSV**.
2. Specify the number of rows to view. Each row in the CSV file corresponds to one Salesforce record. The default is 1,000.
3. To view a specific CSV file, click **Open CSV**. To view the last success file, click **Open Success**. To view the last error file, click **Open Error**.
4. To open the file in an external program, such as Excel, click **Open in External Program**.

   The success file contains all the successfully loaded records. The file includes a column with the newly generated record IDs. The error file contains all the rejected records. The file has a column that describes why the load failed.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Note: If the object you are exporting has a column named "success" or "error," your output file columns could display incorrect information. To avoid this problem, rename the columns.

5. To return to the CSV Chooser window, click **Close**. To exit the window, click **OK**.

Note: To generate success files when exporting data, select `Generate status files for exports`. For more information, see Configure Data Loader on page 8.

Data Import Dates

Limits for importing data with Data Loader.

The following limits apply to data imported using Data Loader.

Only dates within a certain range are valid. The earliest valid date is 1700-01-01T00:00:00Z GMT, or just after midnight on January 1, 1700. The latest valid date is 4000-12-31T00:00:00Z GMT, or just after midnight on December 31, 4000. These values are offset by your time zone. For example, in the Pacific time zone, the earliest valid date is 1699-12-31T16:00:00, or 4:00 PM on December 31, 1699.

When using Data Loader version 28.0 and later, the maximum field size for imported CSV files is 32,000 characters.

View the Data Loader Log File

If you need to investigate a problem with Data Loader, or if requested by Salesforce Customer Support, you can access log files that track the operations and network connections made by Data Loader.

The log file, `sdl.log`, contains a detailed chronological list of Data Loader log entries. Log entries marked "INFO" are procedural items, such as logging in to and out of Salesforce. Log entries marked "ERROR" are problems such as a submitted record missing a required field. The log file can be opened with commonly available text editor programs, such as Microsoft Notepad.

If you are using Data Loader for Windows, view the log file by entering *`%TEMP%\sdl.log`* in either the Run dialog or the Windows Explorer address bar.

If you are using Data Loader for Mac OSX, view the log file by opening terminal and entering *`open $TMPDIR/sdl.log`*.

If you are having login issues from the command line, ensure that the password provided in the configuration parameters is encrypted. If you are having login issues from the UI, you may need to obtain a new security token.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Configure the Data Loader Log File

You can customize the Data Loader log file for advanced troubleshooting and tracking.

The `log-conf.xml` file is included with the Data Loader installer version 35.0.

- In Windows, the `log-conf.xml` file is at `C:\Users\{userName}\dataloader\version\configs`
- In macOS, the `log-conf.xml` file is at `/Users/{userName}/dataloader/version/configs`

Change `@LOG_LEVEL@` to `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, or `FATAL` for the needed level of log tracking. For Log4J log levels, see `https://logging.apache.org/log4j/2.0/manual/architecture.html`.

To implement enhanced logging, use a copy of `log-conf.xml`.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

CHAPTER 6 Running in Batch Mode (Windows Only)

In this chapter ...

- Installed Directories and Files
- Encrypt from the Command Line
- Upgrade Your Batch Mode Interface
- Run Batch File With Windows Command-Line Interface
- Configure Batch Processes
- Data Loader Process Configuration Parameters
- Data Loader Command-Line Operations
- Configure Database Access
- Map Columns
- Run Individual Batch Processes

You can run Data Loader in batch mode from the Windows command line. Batch mode runs a series of Data Loader processes in a certain order using a batch file. You can rerun the same sequence of processes using an executable .bat file.

Note: The Data Loader command-line interface is supported for Windows only.

Note: If you're batch mode from the command line with a version earlier than 8.0, see Upgrade Your Batch Mode Interface on page 27.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Installed Directories and Files

In version 8.0 and later, installing the Data Loader adds several directories under the installation directory that are needed for automated batch processing.

Note: The Data Loader command-line interface is supported for Windows only.

bin

Contains the batch files `encrypt.bat` for encrypting passwords and `process.bat` for running batch processes.

For information on running Data Loader from the command-line, see Run Batch File With Windows Command-Line Interface on page 27.

configs

The default configuration directory. Contains the configuration files `config.properties`, `Loader.class`, and `log-conf.xml`.

The `config.properties` file that is generated when you modify the Settings dialog in the graphical user interface is located at `C:\Users\{userName}\dataloader\version\configs`.

Data Loader runs the operation, file, or map listed in the configuration file that you specify. If you don't specify a configuration directory, the current directory is used.

samples

Contains subdirectories of sample files for reference.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

File Path Convention

The file paths provided in these topics start one level below the installation directory. For example, `\configs` means `C:\Users\{userName}\dataloader\version\configs`, provided you accepted the default installation directory. If you installed the program in a different location, use that directory path.

Encrypt from the Command Line

Data Loader offers an encryption utility to secure passwords specified in configuration files. This utility is used to encrypt passwords, but data that you transmit using Data Loader is not encrypted.

Note: The Data Loader command-line interface is supported for Windows only.

When running Data Loader in batch mode from the command line, you must encrypt the following configuration parameters:

- `sfdc.password`
- `sfdc.proxyPassword`

1. Open a command prompt, and navigate to the `bin` subfolder of your Data Loader installation folder.
2. Run `encrypt.bat`.
3. At the command line, follow the prompts provided to execute the following actions.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Generate a key: `-k [path to key file]`
Generates a key file, and saves it in `%userprofile%\.dataloader\dataLoader.key` if the path is not specified. Store this file with care as you use it for encryption and decryption.

Encrypt text: `-e <plain text> <path to key file>`
Generates an encrypted version of the text. Provide a key file for the encryption.

Decrypt text: `-d <encrypted text> <path to key file>`
Decrypts the text using the key file.

Upgrade Your Batch Mode Interface

Note: The Data Loader command-line interface is supported for Windows only.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

The batch mode interface in Data Loader versions 8.0 and later aren't backward-compatible with earlier versions. If you're using a version earlier than 8.0 to run batch processes, your options are as follows:

Maintain the old version for batch use
Do not uninstall your old version of Data Loader. Continue to use that version for batch processes. You can't take advantage of newer features such as database connectivity, but your integrations will continue to work. Optionally, install the new version alongside the old version and dedicate the old version solely to batch processes.

Generate a new config.properties file from the new GUI
If you originally generated your `config.properties` file from the graphical user interface, use the new version to set the same properties and generate a new file. Use this new file with the new batch mode interface. For more information, see Run Batch File With Windows Command-Line Interface on page 27.

Manually update your config.properties file
If your old `config.properties` file was created manually, you must manually update it for the new version. For more information, see Installed Directories and Files on page 26.

Run Batch File With Windows Command-Line Interface

For automated batch operations, such as nightly scheduled loads and extractions, run Data Loader from the Windows command-line.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Note: The Data Loader command-line interface is supported for Windows only.

Include your encrypted password in the configuration file to run a batch operation. For more information, see Data Loader Command Line Introduction on page 46 and Encrypt from the Command Line on page 26. From the command line, navigate to the `bin` directory, and enter `process.bat`. This file takes the following parameters.

- The directory containing `config.properties`
- The name of the batch process bean contained in `process-conf.xml`

You use the `process-conf.xml` file to configure batch processing. Set the name of the process in the bean element's id attribute: (for example <bean id="myProcessName">).

You can change parameters at run time by giving *param=value* as program arguments. For example, adding `process.operation=insert` to the command changes the configuration at run time.

The *process.bat* file supports running JAR with up to two parameters. The format for two parameters is:

```
java -cp ..\dataloader-xx.0.0-uber.jar -Dsalesforce.config.dir=<configdir>
com.salesforce.dataloader.process.ProcessRunner <operation> parameterName1=ParemeterValue1
 parameterName2=ParemeterValue2
```

An example of running the JAR with two parameters is:

```
"%JAVA_HOME%\bin\java" -cp ..\dataloader-44.1.0-uber.jar
-Dsalesforce.config.dir="C:\Users\username\dataloaderconfig"
com.salesforce.dataloader.process.ProcessRunner accountInsert
process.outputSuccess=c:\temp\accountInsert_success.csv
process.outputError=c:\temp\accountInsert_error.csv
```

Tip: If you experience login issues in the command-line interface after upgrading Data Loader, try encrypting your password again to solve the problem. For information, see Encrypt from the Command Line on page 26.

Configure Batch Processes

Note: The Data Loader command-line interface is supported for Windows only.

Use `\samples\conf\process-conf.xml` to configure your Data Loader processes, which are represented by ProcessRunner beans. A process should have `ProcessRunner` as the `class` attribute and the following properties set in the configuration file:

name
Sets the name of the ProcessRunner bean. This value is also used as the non-generic thread name and for configuration backing files (see below).

configOverrideMap
A property of type `map` where each entry represents a configuration setting: the key is the setting name; the value is the setting value.

enableLastRunOutput
If set to true (the default), output files containing information about the last run, such as `sendAccountsFile_lastrun.properties`, are generated and saved to the location specified by `lastRunOutputDirectory`. If set to false, the files are not generated or saved.

lastRunOutputDirectory
The directory location where output files containing information about the last run, such as `sendAccountsFile_lastrun.properties`, are written. The default value is `\conf`. If `enableLastRunOutput` is set to false, this value is not used because the files are not generated.

The configuration backing file stores configuration parameter values from the last run for debugging purposes, and is used to load default configuration parameters in `config.properties`. The settings in `configOverrideMap` take precedence over those in the configuration backing file. The configuration backing file is managed programmatically and does not require any manual edits.

For the names and descriptions of available process configuration parameters, see Data Loader Process Configuration Parameters on page 29.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Data Loader Process Configuration Parameters

Note: The Data Loader command-line interface is supported for Windows only.

When running Data Loader from the command line, you can specify the following configuration parameters in the `process-conf.xml` file. In some cases, the parameter is also represented in the UI at **Settings** > **Settings**.

Tip: A sample `process-conf.xml` file is in the `\samples` directory that's installed with Data Loader.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
`dataAccess.readUTF8`	boolean	`Read all CSVs with UTF-8 encoding`	Select this option to force files to open in UTF-8 encoding, even if they were saved in a different format. Sample value: `true`
`dataAccess.writeUTF8`	boolean	`Write all CSVs with UTF-8 encoding`	Select this option to force files to be written in UTF-8 encoding. Sample value: `true`
`dataAccess.name`	string	Not applicable (N/A)	Name of the data source to use, such as a CSV file name. For databases, use the name of the database configuration in `database-conf.xml`. Sample value: `c:\dataloader\data\extractLead.csv`
`dataAccess.readBatchSize`	integer	N/A	Number of records read from the database at a time. The maximum value is 200. Sample value: `50`
`dataAccess.type`	string	N/A	Standard or custom data source type. Standard types are `csvWriter`, `csvRead`, `databaseWrite`, and `databaseRead`. Sample value: `csvWrite`

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			Number of records written to the database at a time. The maximum value is 2,000. Note the implication for a large parameter value: if an error occurs, all records in the batch are rolled back. In contrast, if the value is set to 1, each record is processed individually (not in batch) and errors are specific to a given record. We recommend setting the value to 1 when you need to diagnose problems with writing to a database.
`dataAccess.writeBatchSize`	integer	N/A	Sample value: `500`
`loader.csvComma`	boolean	`Allow comma as a CSV delimiter`	Select this option if your CSV file uses commas to delimit records.
`loader.csvTab`	boolean	`Allow tab as a CSV delimiter`	Select this option if your CSV file uses tab characters to delimit records.
`loader.csvOther`	boolean	`Allow other characters as CSV delimiters`	Select this option if your CSV file uses a character other than a comma or tab to delimit records.
`loader.csvOtherValue`	string	`Other delimiters (enter multiple values with no separator; for example, !+?)`	The characters in this field are used only if the **Allow other characters as CSV delimiters** option is selected. For example, if you use the | (pipe) character to delimit data records, enter that character in this field.
`process.enableExtractStatusOutput`	boolean	`Generate status files for exports`	Select this option to generate success and error files when exporting data. Sample value: `true`
`process.enableLastRunOutput`	boolean	N/A	When running Data Loader in batch mode, you can disable the generation of output files such as

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			`sendAccountsFile_lastRun.properties`. Files of this type are saved by default to the `conf` directory. To stop the writing of these files, set this option to `false`. Alternatively, you can change the location of the directory where these files are saved, using `process.lastRunOutputDirectory`. Sample value: `true`
`process.encryptionKeyFile`	string (file name)	N/A	Name of the file that contains the encryption key. This parameter is required in Data Loader version 43.0 and later. See Encrypt from the Command Line on page 26. Sample value: `c:\Users\{user}\.dataloader\dataloader.key`
`process.initialLastRunDate`	date	N/A	The initial setting for the `process.lastRunDate` parameter, which can be used in a SQL string and is automatically updated when a process has run successfully. For an explanation of the date format syntax, see Date Formats on page 15. Format must be `yyyy-MM-ddTHH:mm:ss.SSS+/-HHmm`. For example: 2006-04-13T13:50:32.423-0700
`process.lastRunOutputDirectory`	string (directory)	N/A	When running Data Loader in batch mode, you can change the location where output files such as `sendAccountsFile_lastRun.properties` are written. Files of this type are saved by default to the `\conf` directory. To change the location, change the value of this option to the full path where you want the output files written. Alternatively, you can stop the files from being written, using `process.enableLastRunOutput`.
`process.loadRowToStartAt`	number	`Start at row`	If your last operation failed, you can use this setting to begin where the last successful operation finished. Sample value: `1008`

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
`process.mappingFile`	string (file name)	N/A	Name of the field mapping file to use. See Map Columns on page 42. Sample value: `c:\dataloader\conf\accountExtractMap.sdl`
`process.operation`	string	N/A	The operation to perform. See Data Loader Command-Line Operations on page 37. Sample value: `extract`
`process.statusOutputDirectory`	string (directory)	N/A	The directory where "success" and "error" output files are saved. The file names are automatically generated for each operation unless you specify otherwise in `process-conf.xml`. Sample value: `c:\dataloader\status`
`process.outputError`	string (file name)	N/A	The name of the CSV file that stores error data from the last operation. Sample value: `c:\dataloader\status\myProcessErrors.csv`
`process.outputSuccess`	string (file name)	N/A	The name of the CSV file that stores success data from the last operation. See also `process.enableExtractStatusOutput` on page 30. Sample value: `c:\dataloader\status\myProcessSuccesses.csv`
`process.useEuropeanDates`	boolean	`Use European date format`	Select this option to support the date formats `dd/MM/yyyy` and `dd/MM/yyyy HH:mm:ss`. Sample value: `true`
`sfdc.assignmentRule`	string	`Assignment rule`	Specify the ID of the assignment rule to use for inserts, updates, and upserts. This option applies to inserts, updates, and upserts on cases and leads. With the original territory management feature, it also applies to updates on accounts if your organization has territory assignment rules on accounts. The assignment rule overrides `Owner` values in your CSV file. Sample value: `03Mc00000026J7w`

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
sfdc.bulkApiCheckStatusInterval	integer	N/A	The number of milliseconds to wait between successive checks to determine if the asynchronous Bulk API operation is complete or how many records have been processed. See also sfdc.useBulkApi. We recommend a value of 5000. Sample value: 5000
sfdc.bulkApiSerialMode	boolean	Enable serial mode for Bulk API	To use serial processing instead of parallel processing for Bulk API, select this option. Processing in parallel can cause database contention. When contention is severe, the load can fail. Serial mode processes batches one at a time, however it can increase the processing time for a load. See also sfdc.useBulkApi. Sample value: false
sfdc.bulkApiZipContent	boolean	Upload Bulk API Batch as Zip File	Select this option to use Bulk API to upload zip files containing binary attachments, such as Attachment records or Salesforce CRM Content. See also sfdc.useBulkApi. Sample value: true
sfdc.connectionTimeoutSecs	integer	N/A	The number of seconds to wait for a connection during API calls. Sample value: 60
sfdc.debugMessages	boolean	N/A	If true, enables SOAP message debugging. By default, messages are sent to STDOUT unless you specify an alternate location in sfdc.debugMessagesFile. Sample value: false
sfdc.debugMessagesFile	string (file name)	N/A	See process.enableExtractStatusOutput on page 30. Stores SOAP messages sent to or from Salesforce. As messages are sent or received, they are appended to the end of the file. As the file does not have a size limit, monitor your available disk storage appropriately. Sample value: \lexiloader\status\sfdcSoapTrace.log

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
sfdc.enableRetries	boolean	N/A	If true, enables repeated attempts to connect to Salesforce servers. See sfdc.maxRetries on page 35 and sfdc.minRetrySleepSecs on page 35. Sample value: true
sfdc.endpoint	URL	Server host	Enter the URL of the Salesforce server with which you want to communicate. For example, if you are loading data into a sandbox, change the URL to https://test.salesforce.com. Sample production value: https://login.salesforce.com/services/Soap/u/46.0
sfdc.entity	string	N/A	The Salesforce object used in the operation. Sample value: Lead
sfdc.externalIdField	string	N/A	Used in upsert operations; specifies the custom field with the "External ID" attribute that is used as a unique identifier for data matching. Sample value: LegacySKU__c
sfdc.extractionRequestSize	integer	Query request size	In a single export or query operation, records are returned from Salesforce in increments of this size. Larger values can improve performance but use more memory on the client. Sample value: 500
sfdc.extractionSOQL	string	N/A	The SOQL query for the data export. Sample value: SELECT Id, LastName, FirstName, Rating, AnnualRevenue, OwnerId FROM Lead
sfdc.insertNulls	boolean	Insert null values	Select this option to insert blank mapped values as null values during data operations. When you are updating records, this option instructs Data Loader to overwrite existing data in mapped fields. Sample value: false
sfdc.loadBatchSize	integer	Batch size	In a single insert, update, upsert, or delete operation, records moving to or from Salesforce are processed in increments of this size. The maximum is 200 records. We recommend a value between 50 and 100.

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			Sample value: `100`
`sfdc.maxRetries`	integer	N/A	The maximum number of repeated attempts to connect to Salesforce. See `sfdc.enableRetries` on page 34. Sample value: `3`
`sfdc.minRetrySleepSecs`	integer	N/A	The minimum number of seconds to wait between connection retries. The wait time increases with each try. See `sfdc.enableRetries` on page 34. Sample value: `2`
`sfdc.noCompression`	boolean	`Compression`	Compression enhances the performance of Data Loader and is turned on by default. You might want to disable compression when debugging the underlying SOAP messages. To turn off compression, enable this option. Sample value: `false`
`sfdc.password`	encrypted string	N/A	An encrypted Salesforce password that corresponds to the username provided in `sfdc.username`. This parameter is required in Data Loader version 43.0 and later. See also Encrypt from the Command Line on page 26. Sample value: `4285b36161c65a22`
`sfdc.proxyHost`	URL	`Proxy host`	The host name of the proxy server, if applicable. Sample value: `http://myproxy.internal.company.com`
`sfdc.proxyPassword`	encrypted string	`Proxy password`	An encrypted password that corresponds to the proxy username provided in `sfdc.proxyUsername`. See also Encrypt from the Command Line on page 26. Sample value: `4285b36161c65a22`
`sfdc.proxyPort`	integer	`Proxy port`	The proxy server port. Sample value: `8000`
`sfdc.proxyUsername`	string	`Proxy username`	The username for proxy server authentication. Sample value: `jane.doe`

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
`sfdc.resetUrlOnLogin`	boolean	`Reset URL on Login`	By default, Salesforce resets the URL after login to the one specified in `sfdc.endpoint`. To turn off this automatic reset, disable this option by setting it to `false`. Valid values: `true` (default), `false`
`sfdc.timeoutSecs`	integer	`Timeout`	Specify how many seconds Data Loader waits to receive a response back from the server before returning an error for the request. Sample value: `540`
`sfdc.timezone`	string	`Time Zone`	If a date value does not include a time zone, this value is used. • If no value is specified, the time zone of the computer where Data Loader is installed is used. • If an incorrect value is entered, GMT is used as the time zone and this fact is noted in the Data Loader log. Valid values are any time zone identifier which can be passed to the Java `getTimeZone(java.lang.String)` method. The value can be a full name such as `America/Los_Angeles`, or a custom ID such as `GMT-8:00`. You can retrieve the default value by running the `TimeZone.getDefault()` method in Java. This value is the time zone on the computer where Data Loader is installed.
`sfdc.truncateFields`	boolean	`Allow field truncation`	Select this option to truncate data in the following types of fields when loading that data into Salesforce: Email, Multi-select Picklist, Phone, Picklist, Text, and Text (Encrypted). In Data Loader versions 14.0 and earlier, Data Loader truncates values for fields of those types if they are too large. In Data Loader version 15.0 and later, the load operation fails if a value is specified that is too large. Selecting this option allows you to specify that the previous behavior, truncation, be used instead of the new behavior in Data Loader versions 15.0 and later.

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			This option is selected by default and has no effect in versions 14.0 and earlier. This option is not available if the `Use Bulk API` option is selected. In that case, the load operation fails for the row if a value is specified that is too large for the field. Sample value: `true`
`sfdc.useBulkApi`	boolean	`Use Bulk API`	Select this option to use Bulk API to insert, update, upsert, delete, and hard-delete records. Bulk API is optimized to load or delete many records asynchronously. It's faster than the default SOAP-based API due to parallel processing and fewer network round-trips. See also `sfdc.bulkApiSerialMode`. Sample value: `true`
`sfdc.username`	string	N/A	Salesforce username. See `sfdc.password`. Sample value: `jdoe@mycompany.com`

Data Loader Command-Line Operations

Note: The Data Loader command-line interface is supported for Windows only.

When running Data Loader in batch mode from the command line, several operations are supported. An operation represents the flow of data between Salesforce and an external data source, such as a CSV file or database. You can use the following operations. Enter values in the `process.operation` parameter in lowercase

Operation	Description
extract	Uses the Salesforce Object Query Language to export a set of records from Salesforce. The exported data is written to a data source. Soft-deleted records are not included.
extract all	Uses SOQL to export a set of records from Salesforce, including existing and soft-deleted records. The exported data is written to a data source.
insert	Loads data from a data source into Salesforce as new records.
update	Loads data from a data source into Salesforce, and updates existing records with matching ID fields.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Operation	Description
upsert	Loads data from a data source into Salesforce. Existing records with a matching custom external ID field are updated. Records without matches are inserted as new records.
delete	Loads data from a data source into Salesforce, and deletes existing records with matching ID fields. Deleted records are moved to the Recycle Bin.
hard delete	Loads data from a data source into Salesforce, and deletes existing records with matching ID fields without first storing them in the Recycle Bin.

Configure Database Access

Note: The Data Loader command-line interface is supported for Windows only.

When you run Data Loader in batch mode from the command line, use `\samples\conf\database-conf.xml` to configure database access objects, which you use to extract data directly from a database.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

DatabaseConfig Bean

The top-level database configuration object is the `DatabaseConfig` bean, which has the following properties:

sqlConfig
The SQL configuration bean for the data access object that interacts with a database.

dataSource
The bean that acts as database driver and authenticator. It must refer to an implementation of `javax.sql.DataSource` such as `org.apache.commons.dbcp.BasicDataSource`.

The following code is an example of a DatabaseConfig bean:

```
<bean id="AccountInsert"
    class="com.salesforce.dataloader.dao.database.DatabaseConfig"
    singleton="true">
    <property name="sqlConfig" ref="accountInsertSql"/>
</bean>
```

DataSource

The `DataSource` bean sets the physical information needed for database connections. It contains the following properties:

driverClassName
The fully qualified name of the implementation of a JDBC driver.

url
The string for physically connecting to the database.

username
The username for logging in to the database.

password
The password for logging in to the database.

Depending on your implementation, additional information may be required. For example, use `org.apache.commons.dbcp.BasicDataSource` when database connections are pooled.

The following code is an example of a DataSource bean:

```
<bean id="oracleRepDataSource"
    class="org.apache.commons.dbcp.BasicDataSource"
    destroy-method="close">
    <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"/>
    <property name="url" value="jdbc:oracle:thin:@myserver.salesforce.com:1521:TEST"/>
    <property name="username" value="test"/>
    <property name="password" value="test"/>
</bean>
```

Versions of Data Loader from API version 25.0 onwards do not come with an Oracle JDBC driver. Using Data Loader to connect to an Oracle data source without a JDBC driver installed will result in a "Cannot load JDBC driver class" error. To add the Oracle JDBC driver to Data Loader:

- Download the latest JDBC driver from `http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html`.
- Copy the JDBC .jar file to *`data loader install folder`*`/java/bin`.

Spring Framework

Note: The Data Loader command-line interface is supported for Windows only.

The Data Loader configuration files are based on the Spring Framework, which is an open-source, full-stack Java/J2EE application framework.

The Spring Framework allows you to use XML files to configure beans. Each bean represents an instance of an object; the parameters correspond to each object's setter methods. A typical bean has the following attributes:

id
Uniquely identifies the bean to `XmlBeanFactory`, which is the class that gets objects from an XML configuration file.

class
Specifies the implementation class for the bean instance.

For more information on the Spring Framework, see the official documentation and the support forums. Note that Salesforce cannot guarantee the availability or accuracy of external websites.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Data Access Objects

Note: The Data Loader command-line interface is supported for Windows only.

When running Data Loader in batch mode from the command line, several data access objects are supported. A data access object allows access to an external data source outside of Salesforce. They can implement a read interface (`DataReader`), a write interface (`DataWriter`), or both. See the following list of object names and descriptions.

csvRead
Allows the reading of a comma or tab-delimited file. There should be a header row at the top of the file that describes each column.

csvWrite
Allows writing to a comma-delimited file. A header row is added to the top of the file based on the column list provided by the caller.

databaseRead
Allows the reading of a database. Use `database-conf.xml` to configure database access.

databaseWrite
Allows writing to a database. Use `database-conf.xml` to configure database access.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

SQL Configuration

Note: The Data Loader command-line interface is supported for Windows only.

When running Data Loader in batch mode from the command line, the `SqlConfig` class contains configuration parameters for accessing specific data in the database. As shown in the code samples below, queries and inserts are different but very similar. The bean must be of type `com.salesforce.dataloader.dao.database.SqlConfig` and have the following properties:

`sqlString`
The SQL code to be used by the data access object.

The SQL can contain replacement parameters that make the string dependent on configuration or operation variables. Replacement parameters must be delimited on both sides by "@" characters. For example, `@process.lastRunDate@`.

`sqlParams`
A property of type `map` that contains descriptions of the replacement parameters specified in `sqlString`. Each entry represents one replacement parameter: the key is the replacement parameter's name, the value is the fully qualified Java type to be used when the parameter is set on the SQL statement. Note that "java.sql" types are sometimes required, such as `java.sql.Date` instead of `java.util.Date`. For more information, see the official JDBC API documentation.

`columnNames`
Used when queries (`SELECT` statements) return a JDBC `ResultSet`. Contains column names for the data outputted by executing the SQL. The column names are used to access and return the output to the caller of the `DataReader` interface.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

SQL Query Bean Example

```
<bean id="accountMasterSql"
    class="com.salesforce.dataloader.dao.database.SqlConfig"
```

```
    singleton="true">
    <property name="sqlString"/>
        <value>
            SELECT distinct
                '012x00000000Ij7' recordTypeId,
                accounts.account_number,
                org.organization_name,
                concat (concat(parties.address1, ' '), parties.address2) billing_address,

                locs.city,
                locs.postal_code,
                locs.state,
                locs.country,
                parties.sic_code
            from
                ar.hz_cust_accounts accounts,
                ar.hz_organization_profiles org,
                ar.hz_parties parties,
                ar.hz_party_sites party_sites,
                ar.hz_locations locs
            where
                accounts.PARTY_ID = org.PARTY_ID
                and parties.PARTY_ID = accounts.PARTY_ID
                and party_sites.PARTY_ID = accounts.PARTY_ID
                and locs.LOCATION_ID = party_sites.LOCATION_ID
                and (locs.last_update_date > @process.lastRunDate@ OR
accounts.last_update_date > @process.lastRunDate@
        </value>
    </property>
    <property name="columNames">
        <list>
            <value>recordTypeId</value>
            <value>account_number</value>
            <value>organization_name</value>
            <value>billing_address</value>
            <value>city</value>
            <value>postal_code</value>
            <value>state</value>
            <value>country</value>
            <value>sic_code</value>
        </list>
    </property>
    <property name="sqlParams">
        <map>
            <entry key="process.lastRunDate" value="java.sql.Date"/>
        </map>
    </property>
</bean>
```

SQL Insert Bean Example

```
<bean id="partiesInsertSql"
    class="com.salesforce.dataloader.dao.database.SqlConfig"
```

```
    singleton="true">
    <property name="sqlString"/>
        <value>
            INSERT INTO REP.INT_PARTIES (
            BILLING_ADDRESS, SIC_CODE)
            VALUES (@billing_address@, @sic_code@)
        </value>
    </property>
    <property name="sqlParams"/>
        <map>
            <entry key="billing_address" value="java.lang.String"/>
            <entry key="sic_code" value="java.lang.String"/>
        </map>
    </property>
</bean>
```

Map Columns

Note: The Data Loader command-line interface is supported for Windows only.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

When running Data Loader in batch mode from the command line, you must create a properties file that maps values between Salesforce and data access objects.

1. Create a new mapping file and give it an extension of `.sdl`.
2. Observe the following syntax:
   - On each line, pair a data source with its destination.
   - In an import file, put the data source on the left, an equals sign (=) as a separator, and the destination on the right. In an export file, put the destination on the left, an equals sign (=) as a separator, and the data source on the right.
   - Data sources can be either column names or constants. Surround constants with double quotation marks, as in "sampleconstant". Values without quotation marks are treated as column names.
   - Destinations must be column names.
   - You may map constants by surrounding them with double quotation marks, as in:

     ```
     "Canada"=BillingCountry
     ```

3. In your configuration file, use the parameter `process.mappingFile` to specify the name of your mapping file.

Note: If your field name contains a space, you must escape the space by prepending it with a backslash (\). For example:

```
Account\ Name=Name
```

Column Mapping Example for Data Insert

The Salesforce fields are on the right.

```
SLA__C=SLA__c
BILLINGCITY=BillingCity
SYSTEMMODSTAMP=
```

```
OWNERID=OwnerId
CUSTOMERPRIORITY__C=CustomerPriority__c
ANNUALREVENUE=AnnualRevenue
DESCRIPTION=Description
BILLINGSTREET=BillingStreet
SHIPPINGSTATE=ShippingState
```

Column Mapping Example for Data Export

The Salesforce fields are on the left.

```
Id=account_number
Name=name
Phone=phone
```

Column Mapping for Constant Values

Data Loader supports the ability to assign constants to fields when you insert, update, and export data. If you have a field that should contain the same value for each record, you specify that constant in the `.sdl` mapping file instead of specifying the field and value in the CSV file or the export query.

The constant must be enclosed in double quotation marks. For example, if you're importing data, the syntax is `"constantvalue"=field1`.

If you have multiple fields that should contain the same value, you must specify the constant and the field names separated by commas. For example, if you're importing data, the syntax would be `"constantvalue"=field1, field2`.

Here's an example of an `.sdl` file for inserting data. The Salesforce fields are on the right. The first two lines map a data source to a destination field, and the last three lines map a constant to a destination field.

```
Name=Name
NumEmployees=NumberOfEmployees
"Aerospace"=Industry
"California"=BillingState, ShippingState
"New"=Customer_Type__c
```

A constant must contain at least one alphanumeric character.

Note: If you specify a constant value that contains spaces, you must escape the spaces by prepending each with a backslash (\). For example:

```
"Food\ &\ Beverage"=Industry
```

Run Individual Batch Processes

You can run one batch process at a time.

Note: The Data Loader command-line interface is supported for Windows only.

To start an individual batch process, use `\bin\process.bat`. The command-line requires the following parameters.

Configuration directory

The default is `\conf`.

To use an alternate directory, create a directory and add the following files to it.

- If your process is not interactive, copy `process-conf.xml` from `\samples\conf`.
- If your process requires database connectivity, copy `database-conf.xml` from `\samples\conf`.
- Copy `config.properties` from `\conf`.

Process name

The name of the ProcessRunner bean from `\samples\conf\process-conf.xml`.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Process Example

```
process ../conf accountMasterProcess
```

Note: To view tips and instructions, add *`-help`* to the command contained in `process.bat`.

Note: You can configure external process launchers, such as the Windows XP Scheduled Task Wizard, to run processes on a schedule.

CHAPTER 7 Command-Line Quick Start (Windows Only)

In this chapter ...

- Data Loader Command Line Introduction
- Prerequisites
- Step One: Create the Encryption Key File
- Step Two: Create the Encrypted Password
- Step Three: Create the Field Mapping File
- Step Four: Create the Configuration File
- Step Five: Import the Data

Note: The Data Loader command-line interface is supported for Windows only.

This quick start shows you how to use the Data Loader command-line functionality to import data.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Data Loader Command Line Introduction

Note: The Data Loader command-line interface is supported for Windows only.

In addition to using Data Loader interactively to import and export data, you can run it from the command line. You can use commands to automate the import and export of data.

This quick start shows you how to use the Data Loader command-line functionality to import data. Follow these steps.

- Step 1: Create the encryption key
- Step 2: Create the encrypted password for your login username
- Step 3: Create the Field Mapping File
- Step 4: Create a `process-conf.xml` file that contains the import configuration settings
- Step 5: Run the process and import the data

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

SEE ALSO:

Prerequisites

Prerequisites

Note: The Data Loader command-line interface is supported for Windows only.

To step through this quick start requires the following:

- Data Loader installed on the computer that runs the command-line process.
- The Java Runtime Environment (JRE) installed on the computer that runs the command-line process.
- Familiarity with importing and exporting data by using the Data Loader interactively through the user interface. This makes it easier to understand how the command-line functionality works.

Tip: When you install Data Loader, sample files are installed in the `samples` directory under the user's directory, for example, `C:\Users\{userName}\dataloader\`*`version`*`\samples\`. Examples of files that are used in this quick start are in the `\samples\conf` directory.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

SEE ALSO:

Data Loader Command Line Introduction

Step One: Create the Encryption Key File

Step One: Create the Encryption Key File

Note: The Data Loader command-line interface is supported for Windows only.

When you use Data Loader from the command line, there's no user interface. Therefore, you provide the information that you would enter in the user interface in a text file named `process-conf.xml`. For example, you add the username and password that Data Loader uses to log in to Salesforce. The password must be encrypted before you add it to the `process-conf.xml` file, and creating the key is the first step in that process.

EDITIONS

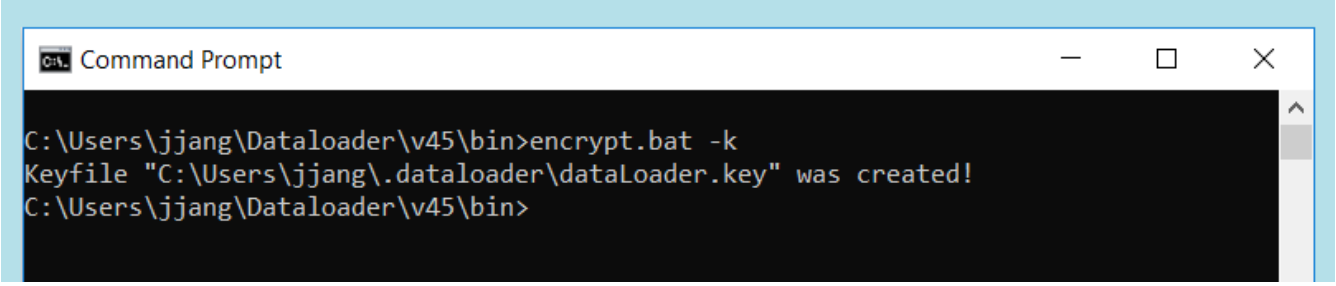
Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

1. Open a command prompt window by selecting **Start** > **All Programs** > **Accessories** > **Command Prompt**. Alternatively, you can click **Start** > **Run**, enter *cmd* in the **Open** field, and click **OK**.
2. In the command window, enter *cd* to navigate to the root directory of the drive where Data Loader is installed.
3. Navigate to the Data Loader `\bin` directory by entering this command. Replace the file path with the path from your system.

   `C:\Users\{userName}\dataloader\version\bin`
4. Create an encryption key file by entering the following command. Replace [*path to key file*] with the key file path.

   encrypt.bat –k [path to key file]



Note: To see a list of command-line options for `encrypt.bat`, enter *encrypt.bat* on the command line.

5. Note the key file path. In this example, the path is `C:\Users\IEUser\.dataloader\dataLoader.key`.

The encryption utility encrypts passwords but not data. HTTPS with TLS 1.0 or later encrypts data transmitted by the Apex Data Loader.

SEE ALSO:

Data Loader Command Line Introduction

Step Two: Create the Encrypted Password

Step Two: Create the Encrypted Password

Note: The Data Loader command-line interface is supported for Windows only.

Create the encrypted password using the key file that you generated in the previous step.

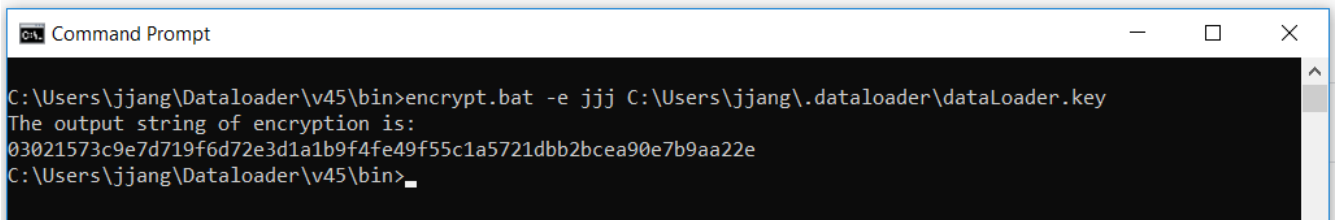
1. In the same command prompt window, enter the following command. Replace *<password>* with the password that you use to log in to Salesforce in Data Loader. Replace *<key file path>* with the file path you created in the previous step.

   ```
   encrypt.bat -e <password> <key file path>
   ```

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions



2. Copy the generated encrypted password. You use this value in a later step.

SEE ALSO:

Data Loader Command Line Introduction

Step Three: Create the Field Mapping File

Step Three: Create the Field Mapping File

Note: The Data Loader command-line interface is supported for Windows only.

In this step, you create a mapping file with an `.sdl` file extension. In each line of the mapping file, pair a data source with its destination.

1. Copy the following to a text file and save it with a name of `accountInsertMap.sdl`. This is a data insert, so the data source is on the left of the equals sign and the destination field is on the right.

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

```
#Mapping values
#Thu May 26 16:19:33 GMT 2011
Name=Name
NumberOfEmployees=NumberOfEmployees
Industry=Industry
```

Tip: For complex mappings, you can use the Data Loader user interface to map source and destination fields and then save those mappings to an `.sdl` file. This is done on the Mapping dialog box by clicking **Save Mapping**.

SEE ALSO:

Data Loader Command Line Introduction

Step Four: Create the Configuration File

Step Four: Create the Configuration File

Note: The Data Loader command-line interface is supported for Windows only.

The `process-conf.xml` file contains the information that Data Loader needs to process the data. Each `<bean>` in the `process-conf.xml` file refers to a single process such as an insert, upsert, or export. Therefore, this file can contain multiple processes. In this step, you edit the file to insert accounts into Salesforce.

1. Make a copy of the `process-conf.xml` file from the `\samples\conf` directory. Be sure to maintain a copy of the original because it contains examples of other types of Data Loader processing such as upserts and exports.
2. Open the file in a text editor, and replace the contents with the following XML:

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

```
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
    <bean id="accountInsert"
        class="com.salesforce.dataloader.process.ProcessRunner"
        singleton="false">
        <description>accountInsert job gets the account record from the CSV file
            and inserts it into Salesforce.</description>
        <property name="name" value="accountInsert"/>
        <property name="configOverrideMap">
            <map>
                <entry key="sfdc.debugMessages" value="true"/>
                <entry key="sfdc.debugMessagesFile"
                    value="C:\DLTest\Log\accountInsertSoapTrace.log"/>
                <entry key="sfdc.endpoint" value="https://servername.salesforce.com"/>
                <entry key="sfdc.username" value="admin@Org.org"/>
                <!--Password below has been encrypted using key file,
                    therefore, it will not work without the key setting:
                    process.encryptionKeyFile.
                    The password is not a valid encrypted value,
                    please generate the real value using the encrypt.bat utility -->
                <entry key="sfdc.password" value="e8a68b73992a7a54"/>
                <entry key="process.encryptionKeyFile"
                    value="c:\Users\{user}\.dataloader\dataLoader.key"/>
                <entry key="sfdc.timeoutSecs" value="600"/>
                <entry key="sfdc.loadBatchSize" value="200"/>
                <entry key="sfdc.entity" value="Account"/>
                <entry key="process.operation" value="insert"/>
```

```
                <entry key="process.mappingFile"
                    value="C:\DLTest\Command Line\Config\accountInsertMap.sdl"/>
                <entry key="dataAccess.name"
                    value="C:\DLTest\In\insertAccounts.csv"/>
                <entry key="process.outputSuccess"
                    value="c:\DLTest\Log\accountInsert_success.csv"/>
                <entry key="process.outputError"
                    value="c:\DLTest\Log\accountInsert_error.csv"/>
                <entry key="dataAccess.type" value="csvRead"/>
                <entry key="process.initialLastRunDate"
                    value="2005-12-01T00:00:00.000-0800"/>
            </map>
        </property>
    </bean>
</beans>
```

3. Modify the following parameters in the `process-conf.xml` file. For more information about the process configuration parameters, see Data Loader Process Configuration Parameters on page 29.

   - `sfdc.endpoint`—Enter the URL of the Salesforce instance for your organization; for example, `https://yourInstance.salesforce.com/`.
   - `sfdc.username`—Enter the username Data Loader uses to log in.
   - `sfdc.password`—Enter the encrypted password value that you created in step 2.
   - `process.mappingFile`—Enter the path and file name of the mapping file.
   - `dataAccess.Name`—Enter the path and file name of the data file that contains the accounts that you want to import.
   - `sfdc.debugMessages`—Currently set to `true` for troubleshooting. Set to `false` after your import is up and running.
   - `sfdc.debugMessagesFile`—Enter the path and file name of the command line log file.
   - `process.outputSuccess`—Enter the path and file name of the success log file.
   - `process.outputError`—Enter the path and file name of the error log file.

   Warning: Use caution when using different XML editors to edit the `process-conf.xml` file. Some editors add XML tags to the beginning and end of the file, which causes the import to fail.

SEE ALSO:

Data Loader Command Line Introduction

Step Five: Import the Data

Step Five: Import the Data

USER PERMISSIONS

To insert records:	Create on the record
To update records:	Edit on the record
To upsert records:	Create or Edit on the record
To delete records:	Delete on the record
To hard delete records:	Delete on the record

EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Note: The Data Loader command-line interface is supported for Windows only.

Now that all the pieces are in place, you can run Data Loader from the command line and insert some new accounts.

1. Copy the following data to a file name `accountInsert.csv`. This is the account data that you import into your organization.

```
Name,Industry,NumberOfEmployees
Dickenson plc,Consulting,120
GenePoint,Biotechnology,265
Express Logistics and Transport,Transportation,12300
Grand Hotels & Resorts Ltd,Hospitality,5600
```

2. In the command prompt window, enter the following command:

   process.bat "<file path to process-conf.xml>" <process name>

   - Replace <file path to process-conf.xml> with the path to the directory containing `process-conf.xml`.
   - Replace <process name> with the process specified in `process-conf.xml`.

   Your command should look something like this:

   process.bat "C:\DLTest\Command Line\Config" accountInsert

   After the process runs, the command prompt window displays success and error messages. You can also check the log files: `insertAccounts_success.csv` and `insertAccounts_error.csv`. After the process runs successfully, the `insertAccounts_success.csv` file contains the records that you imported, along with the ID and status of each record. For more information about the status files, see Review Data Loader Output Files on page 22.

SEE ALSO:

Data Loader Command Line Introduction

APPENDIX A Data Loader Third-Party Licenses

The following third-party licenses are included with the installation of Data Loader:

Technology	Version Number	License
Apache Jakarta Commons BeanUtils	1.7	http://www.apache.org/licenses/LICENSE-2.0
Apache Commons Database Connection Pooling (DBCP)	1.4	http://www.apache.org/licenses/LICENSE-2.0
Apache Log4j	1.2.17	http://www.apache.org/licenses/LICENSE-2.0
Eclipse SWT	4.5.1	http://www.eclipse.org/legal/epl-v10.html
Spring Framework	3.1.1	http://www.apache.org/licenses/LICENSE-2.0.txt

Note: Salesforce is not responsible for the availability or content of third-party websites.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

INDEX

D

Data
    import limits 23
Data Loader
    date formats 15

L

Limits
    data import 23