



Version 29.0: Winter '14

Data Loader Guide



Last updated: January 4, 2014

Table of Contents

Chapter 1: Data Loader Overview.....	1
Chapter 2: When to Use Data Loader.....	2
Installing Data Loader.....	3
Configuring Data Loader.....	4
Data Loader Behavior with Bulk API Enabled.....	6
Configuring the Data Loader to Use the Bulk API.....	7
Uninstalling the Data Loader.....	7
Chapter 3: Using Data Loader.....	8
Data Types Supported by Data Loader.....	9
Exporting Data.....	10
Defining Field Mappings.....	12
Inserting, Updating, or Deleting Data Using Data Loader.....	12
Performing Mass Updates.....	13
Performing Mass Deletes.....	14
Uploading Attachments.....	14
Uploading Content with the Data Loader.....	14
Reviewing Output Files.....	16
Troubleshooting Data Loader Operations.....	16
Chapter 4: Running in Batch Mode.....	18
Understanding Installed Directories and Files.....	19
Encrypting From the Command Line.....	19
Upgrading Your Batch Mode Interface.....	20
Using the Command Line Interface.....	20
Configuring Batch Processes.....	21
Data Loader Process Configuration Parameters.....	21
Data Loader Command Line Operations.....	29
Configuring Database Access.....	29
Spring Framework Overview.....	31
Data Access Objects.....	31
SQL Configuration.....	31
Mapping Columns.....	33
Running Individual Batch Processes.....	34
Chapter 5: Command Line Quick Start.....	36
Introduction.....	37
Prerequisites.....	37
Step One: Create the Encryption Key.....	37
Step Two: Create the Encrypted Password.....	38
Step Three: Create the Field Mapping File.....	39
Step Four: Create the Configuration File.....	39
Step Five: Import the Data.....	41

Appendix A: Data Loader Third-Party Licenses.....42

Index.....43

Chapter 1

Data Loader Overview

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

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 are not appropriate for your locale, use a tab or other delimiter.

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—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
- 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 7

To get started, see the following topics:

- [When to Use Data Loader](#)
- [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

In this chapter ...

- [Installing Data Loader](#)
- [Configuring Data Loader](#)
- [Uninstalling the Data Loader](#)

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

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:

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.com partner or visit the [App Exchange](#) 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, click **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 “Importing Overview” in the Salesforce Help.

Installing Data Loader

User Permissions Needed	
To access the page to download Data Loader:	“Modify All Data”
To use Data Loader:	The appropriate user permission for the operation you are doing, for example, “Create” on accounts to insert new accounts

System Requirements

To use Data Loader, you need:

- Microsoft® Windows® 7 or Windows XP
- 120 MB free disk space
- 256 MB available memory
- Java JRE 1.6 or later (Windows 7 or Windows XP)
- Sun JVM 1.6 or later (Windows 7 or Windows XP)
- Administrator privileges on the machine

Installation Procedure



Warning: Over time, multiple versions of the Data Loader client application have been available for download. Different versions have different entries in the Add or Remove Programs dialog in your Windows Control Panel. Some versions were released with earlier product names such as “AppExchange Data Loader” or “Sforce Data Loader.” You can run *different* versions at the same time on one computer. However, do not install multiple copies of the same version.

The latest version is always available in Salesforce. If you have previously installed the latest version and want to install it again, first remove it from your computer by using the Add or Remove Programs dialog in Windows Control Panel.

1. In the application, from Setup, click **Data Management > Data Loader**.
2. Click **Download the Data Loader** and save the installer to your PC. If you're prompted to run or save the file, click **Run**. If you're then prompted to allow the program to make changes to the computer, click **Yes**.
3. Double-click the downloaded file to launch the InstallShield wizard.
4. Click **Next**.
5. Accept the license agreement and click **Next**.
6. Accept the default installation directory, or click **Change...** to choose another directory. Click **Next**.
7. Click **Install**.
8. Click **Finish**.
9. To start Data Loader, double-click the Data Loader icon on your desktop, or choose **Start > All Programs > salesforce.com > Apex Data Loader > Apex Data Loader**.



Tip:

If you experience login issues in the command line interface after upgrading to a new version of Data Loader, please try re-encrypting your password to solve the problem. For information on the password encryption utility, see [Encrypting From the Command Line](#) on page 19.

If you want to download the source code and make changes, an open source version of Data Loader is available at <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.

Configuring Data Loader

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

1. Start Data Loader by choosing **Start > Programs > salesforce.com > Data Loader > Data Loader**.
2. Choose **Settings > Settings**.
3. Edit the fields as desired:

Field	Description
Batch size	<p>In a single insert, update, upsert, or delete operation, records moving to or from Salesforce are processed in increments of this size. The maximum value is 200. We recommend a value between 50 and 100.</p> <p>The maximum value is 10,000 if the <code>Use Bulk API</code> option is selected.</p>
Insert null values	<p>Select this option to insert blank mapped values as <code>null</code> values during data operations. Note that when you are updating records, this option instructs Data Loader to overwrite any existing data in mapped fields.</p> <p>This option is not available if the <code>Use Bulk API</code> option is selected. Empty field values are ignored when you update records using the Bulk API. To set a field value to <code>null</code> when the <code>Use Bulk API</code> option is selected, use a field value of <code>#N/A</code>.</p>
Assignment rule	<p>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. It also applies to updates on accounts if your organization has territory assignment rules on accounts. The assignment rule overrides <code>Owner</code> values in your CSV file.</p>
Server host	<p>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 <code>https://test.salesforce.com</code>.</p>
Reset URL on Login	<p>By default, Salesforce resets the URL after login to the one specified in <code>Server host</code>. To turn off this automatic reset, disable this option.</p>
Compression	<p>Compression enhances the performance of Data Loader and is turned on by default. You may want to disable compression</p>

Field	Description
	if you need to debug 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. The maximum value is 2,000 records. Larger values may improve performance but use more memory on the client.
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.
Allow field truncation	<p>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).</p> <p>In Data Loader versions 14.0 and earlier, values for fields of those types are truncated by Data Loader 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.</p> <p>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.</p> <p>This option is not available if the <code>Use Bulk API</code> 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.</p>
Use Bulk API	<p>Select this option to use the Bulk API to insert, update, upsert, delete, and hard delete records. The Bulk API is optimized to load or delete a large number of records asynchronously. It's faster than the default SOAP-based API due to parallel processing and fewer network round-trips.</p> <p> Warning: When you select the Hard Delete operation, the deleted records are not stored in the Recycle Bin. Instead, they become immediately eligible for deletion.</p>

Field	Description
Enable serial mode for Bulk API	<p>Select this option to use serial instead of parallel processing for Bulk API. Processing in parallel can cause database contention. When this is severe, the load may fail. Using serial mode guarantees that batches are processed one at a time. Note that using this option may significantly increase the processing time for a load.</p> <p>This option is only available if the Use Bulk API option is selected.</p>
Upload Bulk API Batch as Zip File	<p>Select this option to use Bulk API to upload zip files containing binary attachments, such as Attachment records or Salesforce CRM Content.</p> <p>This option is only available if the Use Bulk API option is selected.</p>
Time Zone	<p>Select this option to specify a default time zone.</p> <p>If a date value does not include a time zone, this value is used.</p> <ul style="list-style-type: none"> 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. <p>Valid values are any time zone identifier which can be passed to the Java <code>getTimeZone(java.lang.String)</code> method. The value can be a full name such as <code>America/Los_Angeles</code>, or a custom ID such as <code>GMT-8:00</code>.</p>
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.

- 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 [Configuring Data Loader](#) on page 4.

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.

Configuring 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. Start Data Loader by choosing **Start > Programs > salesforce.com > Data Loader > 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 this is severe, the load may fail. Using serial mode guarantees that batches are processed one at a time. Note that using this option may significantly increase the processing time for a load.
- **Caution:** When you select the **Hard Delete** operation, the deleted records are not stored in the Recycle Bin. Instead, they become immediately eligible for deletion.

Uninstalling the Data Loader

To uninstall the Data Loader client application:

1. Go to **Start > Control Panel > Add or Remove Programs**.
2. Select the Data Loader program.
3. Click **Remove**. The uninstaller removes the program from your computer.

Chapter 3

Using Data Loader

In this chapter ...

- [Data Types Supported by Data Loader](#)
- [Exporting Data](#)
- [Defining Field Mappings](#)
- [Inserting, Updating, or Deleting Data Using Data Loader](#)
- [Uploading Attachments](#)
- [Uploading Content with the Data Loader](#)
- [Reviewing Output Files](#)
- [Troubleshooting Data Loader Operations](#)

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

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.

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 [Uploading Attachments](#) on page 14 and [Uploading Content with the Data Loader](#) on page 14.

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:

- 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.

Exporting Data

User Permissions Needed	
To export records:	“Read” on the records
To export all records:	“Read” on the records

You can use the Data Loader export wizard to extract data from any Salesforce object. When you export, you can choose to include (**Export All**) or exclude (**Export**) soft-deleted records.

1. Start the Data Loader by choosing **Start > Programs > salesforce.com > Data Loader > Data Loader**.
2. Click **Export** or **Export All**. These commands can also be found in the File menu.
3. Enter your Salesforce username and password. Click **Log in** to log in. After your login completes successfully, click **Next**. (Until you log out or close the program, you will not be 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 XXXXXXXXXXXX, you must enter mypasswordXXXXXXXXXX to log in.

4. Choose an object. For example, select the Account object. If your object name does not display in the default list, check **Show all objects** to see a complete list of objects that you can access. The objects will be listed by localized label name, with developer name noted in parentheses. For object descriptions, see the [SOAP API Developer's Guide](#).
5. Click **Browse...** to select the CSV file to which the data will be exported. You can enter a new file name to create a new file or choose an existing file.

If you select an existing file, the contents of that file are replaced. Click **Yes** to confirm this action, or click **No** to choose another file.

6. Click **Next**.
7. Create a SOQL query for the data export. For example, check `Id` and `Name` in the query fields and click **Finish**. As you follow the next steps, you will see that the CSV viewer displays all the Account names and their IDs. SOQL is the Salesforce Object Query Language that allows you to construct simple but powerful query strings. Similar to the `SELECT` command in SQL, SOQL allows you to 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 data set. If you do not select any conditions, all the data to which you have read access will be 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 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 make use of 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 [Force.com SOQL and SOSL Reference](#).

8. Click **Finish**, then click **Yes** to confirm.
9. A progress information window reports the status of the operation.
10. After the operation completes, a confirmation window summarizes your results. Click **View Extraction** to view the CSV file, or click **OK** to close. For more details, see [Reviewing Output Files](#) on page 16.



Note: Data Loader currently does not support the extraction of attachments. As a workaround, we recommend that you use the weekly export feature in the online application to export attachments.

Defining Field Mappings

Whenever 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 [Inserting, Updating, or Deleting Data Using Data Loader](#) on page 12.

1. To automatically match fields with columns, click **Auto-Match Fields to Columns**. The Data Loader automatically populates the list at the bottom of the window, based on the similarity of field and column names. Note that 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 contents of that file are replaced. Click **Yes** to confirm this action, or click **No** to choose another file.
4. Click **OK** to use your mapping for the current operation.

Inserting, Updating, or Deleting Data Using Data Loader

User Permissions Needed	
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

The insert, update, upsert, delete, and hard delete wizards in Data Loader allow you to add new records, modify existing records, or delete existing records. Note that “upsert” is a combination of inserting and updating. 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, then the record is created as new. When you hard delete records, the deleted records are not stored in the Recycle Bin and become immediately eligible for deletion. For more information, see [Configuring Data Loader](#) on page 4.

1. Start Data Loader by choosing **Start > Programs > salesforce.com > Data Loader > Data Loader**.
2. Click **Insert, Update, Upsert, Delete** or **Hard Delete**. These commands can also be found in the File menu.
3. Enter your Salesforce username and password. Click **Log in** to log in. After your login completes successfully, 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 XXXXXXXXXXXX, 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, check **Show all objects** to see a complete list of the objects that you can access. The objects are listed by localized label name, with developer name noted in parentheses. For object descriptions, see the [Object Reference for Salesforce and Force.com](#).

5. Click **Browse...** to select your CSV file. For example, if you are inserting Account records, you could specify a CSV file named `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:
 - a. Your CSV file must contain a column of ID values for matching against existing records. The column may be either an external ID (a custom field with the “External ID” attribute), or `Id` (the Salesforce record ID). From the drop-down list, select which field to use for matching. If the object has no external ID fields, `Id` is automatically 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 drop-down list. If you make no selection here, 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. Click **Choose an Existing Map** to select an existing field mapping, or click **Create or Edit a Map** to create a new map or modify an existing map. For more details and an example of usage, see [Defining Field Mappings](#) on page 12.
9. Click **Next**.
10. For every operation, the Data Loader generates two unique CSV log files; one file name starts with “success,” while the other starts with “error.” Click **Browse...** to specify a directory for these files.
11. Click **Finish** to perform the operation, and then click **Yes** to confirm.
12. As the operation proceeds, a progress information window reports the status of the data movement.
13. After the operation completes, a confirmation window summarizes your results. Click **View Successes** to view your success file, click **View Errors** to open your errors file, or click **OK** to close. For more information, see [Reviewing Output Files](#) on page 16.

**Tip:**

- If you are updating or deleting large amounts of data, review [Performing Mass Updates](#) and [Performing Mass Deletes](#) for tips and best practices.
- There is a five-minute limit to process 100 records when the Bulk API is enabled. Also, 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 failed, some records could have completed successfully, so you must check the results. If you get a timeout error when loading a file, split your file into smaller files, and try again.

Performing 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 [Inserting, Updating, or Deleting Data Using Data Loader](#) on page 12.
5. After the operation, review your success and error log files. See [Reviewing Output Files](#) on page 16.
6. If you made a mistake, use the backup file to update the records to their previous values.

Performing 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 [Exporting Data](#) on page 10.) 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 [Inserting, Updating, or Deleting Data Using Data Loader](#) on page 12.
4. After the operation, review your success and error log files. See [Reviewing Output Files](#) on page 16.

Uploading Attachments

You can use Data Loader to upload attachments to Salesforce. Before uploading attachments, note the following:

- If you intend to upload via the 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 organization to a target Salesforce organization, begin by requesting a data export for the source organization. On the Schedule Export page, make sure to select the Include Attachments... checkbox, which causes the file Attachment.csv to be included in your export. You can use this CSV file to upload the attachments. For more information on the export service, see “Exporting Backup Data” in the Salesforce Help.

To upload attachments:

1. Confirm that the CSV file you intend to use for attachment importing contains the following required columns (each column represents a Salesforce field):

- `ParentId` - the Salesforce ID of the parent record.
- `Name` - the name of the attachment file, such as `myattachment.jpg`.
- `Body` - the absolute path to the attachment on your local drive.

Ensure that the values in the `Body` column contain the full file name of the attachments as they exist on your computer. For example, if an attachment named `myattachment.jpg` is located on your computer at `C:\Export`, `Body` must specify `C:\Export\myattachment.jpg`. Your CSV file might look like this:

```
ParentId,Name,Body
500300000000VDowAAG,attachment1.jpg,C:\Export\attachment1.gif
701300000000iNHAAY,attachment2.doc,C:\Export\files\attachment2.doc
```

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

2. Proceed with an insert or upsert operation; see [Inserting, Updating, or Deleting Data Using Data Loader](#) on page 12. At the Select data objects step, make sure to select the Show all Salesforce objects checkbox and the Attachment object name in the list.

Uploading 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 via the 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, you must specify the actual 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, you must specify the URL in `ContentUrl`. Do not use `PathOnClient` or `VersionData` to upload links.
- You can't export content using the Data Loader.

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 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` - content type ID.



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

To determine the `RecordTypeId` values for your organization using Data Loader, follow the steps in [Exporting Data](#). Your SOQL query might look like this:

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

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

- Log in to Salesforce.
- Enter this URL in your browser:
`http://instanceName.salesforce.com/soap/ajax/29.0/debugshell.html`. Enter the `instanceName`, such as `na1`, for your organization. You can see the `instanceName` in the URL field of your browser after logging in to Salesforce.
- In the AJAX Toolkit Shell page type:

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

- Press **Enter**.
- Click on the arrows for `recordTypeInfo`s.

All of 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,0050000000000000,058700000004Cd0,012300000008o2sAQG,one
```

2. Upload the CSV file for the ContentVersion object; see [Inserting, Updating, or Deleting Data Using Data Loader](#) on page 12. All documents and links will be available in the specified library.

Reviewing Output Files

After every import or export, Data Loader generates two CSV output files that contain the results of the operation. One file name starts with “success,” while the other starts with “error.” During every export, Data Loader saves the extracted data to a CSV file that you specify in the wizard. Data Loader has a built-in CSV file viewer with which you can open and view these files.

To view output files from a Data Loader operation:

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 1000.
3. To view a CSV file of your choice, click **Open CSV**. To view the last success file, click **Open Success**. To view the last error file, click **Open Error**. The CSV file opens in a new window.
4. Optionally, click **Open in External Program** to open the file in the associated external program, such as Microsoft® Office Excel.

The “success” file contains all of the records that were successfully loaded. In this file, there's a column for the newly generated record IDs. The “error” file contains all of the records that were rejected from the load operation. In this file, there's a column that describes why the load failed.

5. Click **Close** to return to the CSV Chooser window, and then click **OK** to exit the window.



Note: To generate success files when exporting data, select the `Generate status files for exports` setting. For more information, see [Configuring Data Loader](#) on page 4.

Troubleshooting Data Loader Operations

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

sdl.log

Contains a detailed chronological list of Data Loader log entries. Log entries marked “INFO” are procedural items, such as logging in to Salesforce. Log entries marked “ERROR” are problems such as a submitted record missing a required field.

sdl_out.log

A supplemental log that contains additional information not captured in sdl.log. For example, it includes log entries for the creation of proxy server network connections.

These files can be opened with commonly available text editor programs, such as Microsoft Notepad.

You can quickly open these files by entering `%TEMP%\sdl.log` and `%TEMP%\sdl_out.log` in either the Run dialog or the Windows Explorer address bar.

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.

Chapter 4

Running in Batch Mode

In this chapter ...

- [Understanding Installed Directories and Files](#)
- [Encrypting From the Command Line](#)
- [Upgrading Your Batch Mode Interface](#)
- [Using the Command Line Interface](#)
- [Configuring Batch Processes](#)
- [Data Loader Process Configuration Parameters](#)
- [Data Loader Command Line Operations](#)
- [Configuring Database Access](#)
- [Mapping Columns](#)
- [Running Individual Batch Processes](#)

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

You can run Data Loader in batch mode from the command line. See the topics in this section for more information.



Note: If you have used the batch mode from the command line with a version earlier than 8.0, see [Upgrading Your Batch Mode Interface](#) on page 20.

Understanding Installed Directories and Files

In versions 8.0 and later, [installing the Data Loader](#) creates several directories under the installation directory. The following directories are involved in running the program from the command line for automated batch processing:

bin

Contains the batch files `encrypt.bat` for [encrypting passwords](#) and `process.bat` for [running batch processes](#).

For information on running the Data Loader from the command line, see [Using the Command Line Interface](#) on page 20.

conf

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:\Documents and Settings\your Windows username\Application Data\salesforce.com\Data Loader version_number`. You can copy this file to the `conf` installation directory to use it for batch processes.

samples

Contains subdirectories of sample files for reference.

File Path Convention

The file paths provided in these topics start one level below the installation directory. For example, `\bin` means `C:\Program Files\salesforce.com\Data Loader version_number\bin`, provided you accepted the default installation directory. If you installed the program to a different location, please substitute that directory path as appropriate.

Encrypting From the Command Line

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

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

Use the utility described below to perform encryption.

Using the Encryption Utility

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.

1. Run `\bin\encrypt.bat`.
2. At the command line, follow the prompts provided to execute the following actions:

Generate a key

Key text is generated on screen from the text you provide. Carefully copy the key text to a key file, omitting any leading or trailing spaces. The key file can then be used for encryption and decryption.

Encrypt text

Generates an encrypted version of a password or other text. Optionally, you can provide a key file for the encryption. In the configuration file, make sure that the encrypted text is copied precisely and the key file is mentioned.

Verify encrypted text

Given encrypted and decrypted versions of a password, verifies whether the encrypted password provided matches its decrypted version. A success or failure message is printed to the command line.

Upgrading Your Batch Mode Interface

The batch mode interface in Data Loader versions 8.0 and later aren't backwards-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 [Using the Command Line Interface](#) on page 20.

Manually update your config.properties file

If your old `config.properties` file was created manually, then you must manually update it for the new version. For more information, see [Understanding Installed Directories and Files](#) on page 19.

Using the Command Line Interface

For automated batch operations such as nightly scheduled loads and extractions, run Data Loader from the command line. Before running any batch operation, be sure to include your encrypted password in the configuration file. For more information, see [Introduction](#) on page 37 and [Encrypting From the Command Line](#) on page 19. From the command line, navigate to the `bin` directory and type `process.bat`, which takes the following parameters:

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

For more information about using `process.bat`, see [Running Individual Batch Processes](#) on page 34.

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

Data Loader runs whatever operation, file, or map is specified in the configuration file that you specify. If you do not specify a configuration directory, the current directory is used. By default, Data Loader configuration files are installed at the following location:

```
C:\Program Files\salesforce.com\Data Loader version number\conf
```

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">`).

If you want to implement enhanced logging, use a copy of `log-conf.xml`.

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

You can set the minimum and maximum heap size. For example, `-Xms256m -Xmx256m` sets the heap size to 256 MB.



Note: These topics only apply to Data Loader version 8.0 and later.



Tip:

If you experience login issues in the command line interface after upgrading to a new version of Data Loader, please try re-encrypting your password to solve the problem. For information on the password encryption utility, see [Encrypting From the Command Line](#) on page 19.

Configuring Batch Processes

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 21.

Data Loader Process Configuration Parameters

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 graphical user interface at **Settings > Settings**.



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

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
<code>dataAccess.readUTF8</code>	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
<code>dataAccess.writeUTF8</code>	boolean	Write all CSVs with UTF-8 encoding	Select this option to force files to be written in UTF-8 encoding. Sample value: true
<code>dataAccess.name</code>	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 <code>database-conf.xml</code> . Sample value: <code>c:\dataloader\data\extractLead.csv</code>
<code>dataAccess.readBatchSize</code>	integer	N/A	Number of records read from the database at a time. The maximum value is 200. Sample value: 50
<code>dataAccess.type</code>	string	N/A	Standard or custom data source type. Standard types are <code>csvWriter</code> , <code>csvRead</code> , <code>databaseWrite</code> , and <code>databaseRead</code> . Sample value: <code>csvWrite</code>
<code>dataAccess.writeBatchSize</code>	integer	N/A	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. Sample value: 500
<code>process.enableExtractStatusOutput</code>	boolean	Generate status files for exports	Select this option to generate success and error files when exporting data. Sample value: true
<code>process.enableLastRunOutput</code>	boolean	N/A	When running Data Loader in batch mode, you can disable the generation of output files such as <code>sendAccountsFile_lastRun.properties</code> . Files of this type are saved by default to the <code>conf</code>

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			<p>directory. To stop the writing of these files, set this option to <code>false</code>.</p> <p>Alternatively, you can change the location of the directory where these files are saved, using <code>process.lastRunOutputDirectory</code>.</p> <p>Sample value: <code>true</code></p>
<code>process.encryptedKeyFile</code>	string (file name)	N/A	<p>Name of the file that contains the encryption key. See Encrypting From the Command Line on page 19.</p> <p>Sample value: <code>c:\dataloader\conf\my.key</code></p>
<code>process.initialLastRunDate</code>	date	N/A	<p>The initial setting for the <code>process.lastRunDate</code> 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 9.</p> <p>Format must be <code>yyyy-MM-ddTHH:mm:ss.SSS+/-HHmm</code>. For example: <code>2006-04-13T13:50:32.423-0700</code></p>
<code>process.lastRunOutputDirectory</code>	string (directory)	N/A	<p>When running Data Loader in batch mode, you can change the location where output files such as <code>sendAccountsFile_lastRun.properties</code> are written. Files of this type are saved by default to the <code>\conf</code> directory. To change the location, change the value of this option to the full path where the output files should be written.</p> <p>Alternatively, you can stop the files from being written, using <code>process.enableLastRunOutput</code>.</p>
<code>process.loadRowToStartAt</code>	number	Start at row	<p>If your last operation failed, you can use this setting to begin where the last successful operation finished.</p> <p>Sample value: <code>1008</code></p>
<code>process.mappingFile</code>	string (file name)	N/A	<p>Name of the field mapping file to use. See Mapping Columns on page 33.</p> <p>Sample value: <code>c:\dataloader\conf\accountExtractMap.sdl</code></p>

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
<code>process.operation</code>	string	N/A	The operation to perform. See Data Loader Command Line Operations on page 29. Sample value: <code>extract</code>
<code>process.statusOutputDirectory</code>	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 <code>process-conf.xml</code> . Sample value: <code>c:\dataloader\status</code>
<code>process.outputError</code>	string (file name)	N/A	The name of the CSV file that stores error data from the last operation. Sample value: <code>c:\dataloader\status\myProcessErrors.csv</code>
<code>process.outputSuccess</code>	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 22. Sample value: <code>c:\dataloader\status\myProcessSuccesses.csv</code>
<code>process.useEuropeanDates</code>	boolean	Use European date format	Select this option to support the date formats <code>dd/MM/yyyy</code> and <code>dd/MM/yyyy HH:mm:ss</code> . Sample value: <code>true</code>
<code>sfdc.assignmentRule</code>	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. 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: <code>03Mc00000026J7w</code>
<code>sfdc.bulkApiCheckStatusInterval</code>	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: <code>5000</code>

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
<code>sfdc.bulkApiSerialMode</code>	boolean	Enable serial mode for Bulk API	Select this option to use serial instead of parallel processing for Bulk API. Processing in parallel can cause database contention. When this is severe, the load may fail. Using serial mode guarantees that batches are processed one at a time. Note that using this option may significantly increase the processing time for a load. See also sfdc.useBulkApi . Sample value: false
<code>sfdc.bulkApiZipContent</code>	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
<code>sfdc.connectionTimeoutSecs</code>	integer	N/A	The number of seconds to wait for a connection during API calls. Sample value: 60
<code>sfdc.debugMessages</code>	boolean	N/A	If true, enables SOAP message debugging. By default, messages are sent to STDOUT unless you specify an alternate location in <code>sfdc.debugMessagesFile</code> . Sample value: false
<code>sfdc.debugMessagesFile</code>	string (file name)	N/A	See process.enableExtractStatusOutput on page 22. 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, please monitor your available disk storage appropriately. Sample value: <code>\lexiload\status\sfdcSoapTrace.log</code>
<code>sfdc.enableRetries</code>	boolean	N/A	If true, enables repeated attempts to connect to Salesforce servers. See sfdc.maxRetries on page 26 and sfdc.minRetrySleepSecs on page 26. Sample value: true
<code>sfdc.endpoint</code>	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 <code>https://test.salesforce.com</code> .

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			Sample production value: https://login.salesforce.com/services/Soap/u/29.0
<code>sfdc.entity</code>	string	N/A	The Salesforce object used in the operation. Sample value: Lead
<code>sfdc.externalIdField</code>	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
<code>sfdc.extractionRequestSize</code>	integer	Query request size	In a single export or query operation, records are returned from Salesforce in increments of this size. The maximum value is 2,000 records. Larger values may improve performance but use more memory on the client. Sample value: 500
<code>sfdc.extractionSOQL</code>	string	N/A	The SOQL query for the data export. Sample value: SELECT Id, LastName, FirstName, Rating, AnnualRevenue, OwnerId FROM Lead
<code>sfdc.insertNulls</code>	boolean	Insert null values	Select this option to insert blank mapped values as null values during data operations. Note that when you are updating records, this option instructs Data Loader to overwrite any existing data in mapped fields. Sample value: false
<code>sfdc.loadBatchSize</code>	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 value is 200. We recommend a value between 50 and 100. Sample value: 100
<code>sfdc.maxRetries</code>	integer	N/A	The maximum number of repeated attempts to connect to Salesforce. See sfdc.enableRetries on page 25. Sample value: 3
<code>sfdc.minRetrySleepSecs</code>	integer	N/A	The minimum number of seconds to wait between connection retries. The wait time

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			increases with each try. See sfdc.enableRetries on page 25. Sample value: 2
<code>sfdc.noCompression</code>	boolean	Compression	Compression enhances the performance of Data Loader and is turned on by default. You may want to disable compression if you need to debug the underlying SOAP messages. To turn off compression, enable this option. Sample value: false
<code>sfdc.password</code>	encrypted string	N/A	An encrypted Salesforce password that corresponds to the username provided in sfdc.username . See also Encrypting From the Command Line on page 19. Sample value: 4285b36161c65a22
<code>sfdc.proxyHost</code>	URL	Proxy host	The host name of the proxy server, if applicable. Sample value: <code>http://myproxy.internal.company.com</code>
<code>sfdc.proxyPassword</code>	encrypted string	Proxy password	An encrypted password that corresponds to the proxy username provided in sfdc.proxyUsername . See also Encrypting From the Command Line on page 19. Sample value: 4285b36161c65a22
<code>sfdc.proxyPort</code>	integer	Proxy port	The proxy server port. Sample value: 8000
<code>sfdc.proxyUsername</code>	string	Proxy username	The username for proxy server authentication. Sample value: jane.doe
<code>sfdc.resetUrlOnLogin</code>	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
<code>sfdc.timeoutSecs</code>	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

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
<code>sfdc.timezone</code>	string	Time Zone	<p>If a date value does not include a time zone, this value is used.</p> <ul style="list-style-type: none"> 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. <p>Valid values are any time zone identifier which can be passed to the Java <code>getTimeZone(java.lang.String)</code> method. The value can be a full name such as <code>America/Los_Angeles</code>, or a custom ID such as <code>GMT-8:00</code>.</p> <p>You can retrieve the default value by running the <code>TimeZone.getDefault()</code> method in Java. This value is the time zone on the computer where Data Loader is installed.</p>
<code>sfdc.truncateFields</code>	boolean	Allow field truncation	<p>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).</p> <p>In Data Loader versions 14.0 and earlier, values for fields of those types are truncated by Data Loader 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.</p> <p>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.</p> <p>This option is not available if the <code>Use Bulk API</code> 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.</p> <p>Sample value: <code>true</code></p>
<code>sfdc.useBulkApi</code>	boolean	Use Bulk API	<p>Select this option to use the Bulk API to insert, update, upsert, delete, and hard delete records. The Bulk API is optimized to load or delete a large number of records asynchronously. It's faster</p>

Parameter Name	Data Type	Equivalent Option in Settings Dialog	Description
			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

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 a database. See the following list of operation names and descriptions.

Extract

Uses a [Salesforce Object Query Language](#) to export a set of records from Salesforce, then writes the exported data to a data source. Soft-deleted records are not included.

Extract All

Uses a Salesforce Object Query Language to export a set of records from Salesforce, including both existing and soft-deleted records, then writes the exported data 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, where existing records with matching ID fields are updated.

Upsert

Loads data from a data source into Salesforce, where 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, where existing records with matching ID fields are deleted.

Hard Delete

Loads data from a data source into Salesforce, where existing records with matching ID fields are deleted without being stored first in the Recycle Bin.

Configuring Database Access

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.

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 Overview

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.com cannot guarantee the availability or accuracy of external websites.

Data Access Objects

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.

SQL Configuration

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.

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="columnNames">
    <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>
```

Mapping Columns

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
```

Running Individual Batch Processes

To start an individual batch process use `\bin\process.bat`, which requires the following parameters:

A configuration directory

The default is `\conf`.

To use an alternate directory, create a new 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`.

A process name

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

Process Example

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



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

Chapter 5

Command Line Quick Start

In this chapter ...

- [Introduction](#)
- [Prerequisites](#)
- [Step One: Create the Encryption Key](#)
- [Step Two: Create the Encrypted Password](#)
- [Step Three: Create the Field Mapping File](#)
- [Step Four: Create the Configuration File](#)
- [Step Five: Import the Data](#)

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

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

Introduction

Watch a Demo:  [Importing Accounts Using the Data Loader Command Line Interface](#) (5:20 minutes)

In addition to using Data Loader interactively to import and export data, you can also run it from the command line. This enables you 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. You'll 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](#)

See Also:

[Prerequisites](#)

Prerequisites

To step through this quick start, you should have 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. This directory is found below the program directory, for example, `C:\Program Files (x86)\salesforce.com\Apex Data Loader 22.0\samples\`. Examples of files that are used in this quick start can be found in the `\samples\conf` directory.

See Also:

[Introduction](#)

[Step One: Create the Encryption Key](#)

Step One: Create the Encryption Key

When you use Data Loader from the command line, there's no user interface. Therefore, you'll need to provide the information that you would normally enter in the user interface by using a text file named `process-conf.xml`. For example, you'll need to add to this file 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.

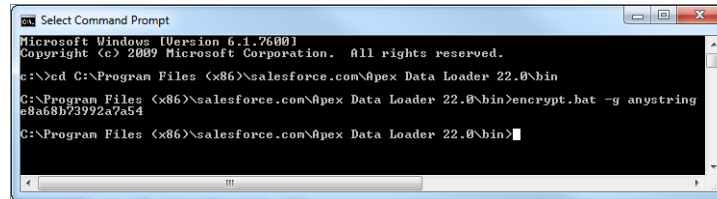
1. Open a command prompt window by clicking **Start > All Programs > Accessories > Command Prompt**. Alternatively, you can click **Start > Run**, enter `cmd` in the **Open** field, and click **OK**.
2. In the command window enter `cd \` to navigate to the root directory of the drive where Data Loader is installed.

3. Navigate to the Data Loader \bin directory by entering this command. Be sure to replace the file path with the path from your system.

```
cd C:\Program Files (x86)\salesforce.com\Apex Data Loader 22.0\bin
```

4. Create an encryption key by entering the following command. Replace <seedtext> with any string.

```
encrypt.bat -g <seedtext>
```



Note: To see a list of command-line options for `encrypt.bat`, type `encrypt.bat` from the command line.

5. Copy the generated key from the command window to a text file named `key.txt` and make a note of the file path. In this example, the generated key is `e8a68b73992a7a54`.



Note: Enabling quick edit mode on a command window can make it easier to copy data to and from the window. To enable quick edit mode, right-click the top of the window and select **Properties**. On the **Options** tab, select **QuickEdit Mode**.

The encryption utility is used to encrypt passwords, but data that you transmit using Data Loader is not encrypted.

See Also:

[Introduction](#)

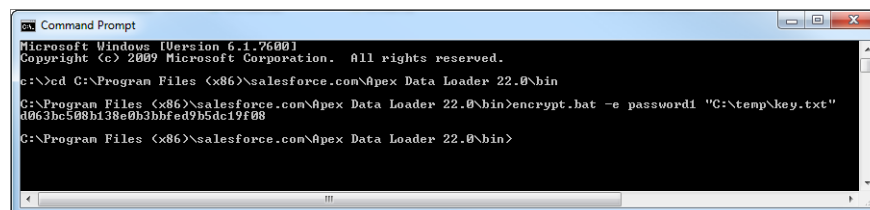
[Step Two: Create the Encrypted Password](#)

Step Two: Create the Encrypted Password

In this step, you'll create the encrypted password using the key that you generated in the previous step.

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

```
encrypt.bat -e <password> "<filepath>\key.txt"
```



2. Copy the encrypted password that is generated by the command. You'll use this value in a later step.

See Also:

[Introduction](#)

[Step Three: Create the Field Mapping File](#)

Step Three: Create the Field Mapping File

The field mapping file associates data sources with destinations. This is a text file, typically with an `.sdl` file extension.

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.

```
#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:

[Introduction](#)

[Step Four: Create the Configuration File](#)

Step Four: Create the Configuration File

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, export, and so on. Therefore, this file can contain multiple processes. In this step, you'll 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:

```
<!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"/>
      </map>
    </property>
  </bean>
</beans>
```

```

        <!--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:\DLTest\Command Line\Config\key.txt"/>
        <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 21.

- `sfdc.endpoint`—Enter the URL of the Salesforce instance for your organization; for example, `https://na1.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 this 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 will cause the import to fail.

See Also:

[Introduction](#)

[Step Five: Import the Data](#)

Step Five: Import the Data

User Permissions Needed	
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

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'll 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 will contain the records that you imported along with the ID and status of each record. For more information about the status files, see [Reviewing Output Files](#) on page 16.

See Also:

[Introduction](#)

Appendix A

Data Loader Third-Party Licenses

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

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

Technology	Version Number	License
Apache Jakarta Commons BeanUtils	1.6	http://www.apache.org/licenses/LICENSE-2.0
Apache Commons Collections	3.1	http://www.apache.org/licenses/LICENSE-2.0
Apache Commons Database Connection Pooling (DBCP)	1.2.1	http://www.apache.org/licenses/LICENSE-2.0
Apache Commons Logging	1.0.3	http://www.apache.org/licenses/LICENSE-1.1
Apache Commons Object Pooling Library	1.2	http://www.apache.org/licenses/LICENSE-2.0
Apache Log4j	1.2.8	http://www.apache.org/licenses/LICENSE-2.0
Eclipse SWT	3.452	http://www.eclipse.org/legal/epl-v10.html
OpenSymphony Quartz Enterprise Job Scheduler	1.5.1	http://www.opensymphony.com/quartz/license.action
Rhino JavaScript for Java	1.6R2	http://www.mozilla.org/MPL/MPL-1.1.txt
Spring Framework	1.2.6	http://www.apache.org/licenses/LICENSE-2.0.txt



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

Index

- A**
- Apex Data Loader
 - See Data Loader [1](#)
- B**
- Bulk API
 - uploading attachments [14](#)
- C**
- Command line
 - configuration file (Data Loader) [39](#)
 - encrypted password (Data Loader) [38](#)
 - encryption key (Data Loader) [37](#)
 - field mapping file (Data Loader) [39](#)
 - importing data (Data Loader) [41](#)
 - introduction (Data Loader) [37](#)
 - prerequisites (Data Loader) [37](#)
 - quick start (Data Loader) [36](#)
- D**
- Data Loader
 - attachments [7](#)
 - batch files [19](#)
 - batch mode [18](#)
 - batch mode parameters [21](#)
 - Bulk API [4](#), [6–7](#), [14](#)
 - column mapping [33](#)
 - command line interface [20](#)
 - command line introduction [37](#)
 - command line operations [29](#)
 - command line quick start [36](#)
 - Data Loader (*continued*)
 - config.properties [21](#)
 - configuration file (command line) [39](#)
 - configuring [4](#), [7](#)
 - configuring batch processes [21](#)
 - data types [9](#)
 - Database Access [29](#)
 - date formats [9](#)
 - encrypted password (command line) [38](#)
 - encryption key (command line) [37](#)
 - field mapping file (command line) [39](#)
 - importing data (command line) [41](#)
 - installed files [19](#)
 - installing [3](#)
 - JDBC Driver [29](#)
 - overview [1](#)
 - password encryption [19](#)
 - prerequisites (command line) [37](#)
 - sample files [19](#)
 - settings [6–7](#)
 - Spring Framework [31](#)
 - starting batch processes [34](#)
 - system requirements [3](#)
 - third-party licenses [42](#)
 - troubleshooting [16](#)
 - uninstalling [7](#)
 - uploading [14](#)
 - uploading attachments [14](#)
 - using [6](#)
 - Using [8](#)
 - when to use [2](#)
- S**
- Spring Framework, see Data Loader [31](#)