Informatica

Informatica® PowerExchange for Salesforce
10.4.1

User Guide for PowerCenter

Informatica PowerExchange for Salesforce User Guide for PowerCenter
10.4.1
December 2020

Publication Date: 2021-03-18

Table of Contents

Preface

Use the *Informatica® PowerExchange® for Salesforce User Guide* to learn how to read from or write to Salesforce by using PowerCenter Client. Learn to create a Salesforce connection, develop mappings, and run sessions in an Informatica domain.

Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

Informatica Network

The Informatica Network is the gateway to many resources, including the Informatica Knowledge Base and Informatica Global Customer Support. To enter the Informatica Network, visit https://network.informatica.com.

As an Informatica Network member, you have the following options:

- Search the Knowledge Base for product resources.
- View product availability information.
- Create and review your support cases.
- Find your local Informatica User Group Network and collaborate with your peers.

Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit https://search.informatica.com. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at KB_Feedback@informatica.com.

Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit https://docs.informatica.com.

If you have questions, comments, or ideas about the product documentation, contact the Informatica Documentation team at infa_documentation@informatica.com.

Informatica Product Availability Matrices

Product Availability Matrices (PAMs) indicate the versions of the operating systems, databases, and types of data sources and targets that a product release supports. You can browse the Informatica PAMs at https://network.informatica.com/community/informatica-network/product-availability-matrices.

Informatica Velocity

Informatica Velocity is a collection of tips and best practices developed by Informatica Professional Services and based on real-world experiences from hundreds of data management projects. Informatica Velocity represents the collective knowledge of Informatica consultants who work with organizations around the world to plan, develop, deploy, and maintain successful data management solutions.

You can find Informatica Velocity resources at http://velocity.informatica.com. If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.

Informatica Marketplace

The Informatica Marketplace is a forum where you can find solutions that extend and enhance your Informatica implementations. Leverage any of the hundreds of solutions from Informatica developers and partners on the Marketplace to improve your productivity and speed up time to implementation on your projects. You can find the Informatica Marketplace at https://marketplace.informatica.com.

Informatica Global Customer Support

You can contact a Global Support Center by telephone or through the Informatica Network.

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:
https://www.informatica.com/services-and-training/customer-success-services/contact-us.html.

To find online support resources on the Informatica Network, visit https://network.informatica.com and select the eSupport option.

CHAPTER 1

Understanding PowerExchange for Salesforce

This chapter includes the following topics:

Understanding PowerExchange for Salesforce Overview

PowerExchange for Salesforce integrates PowerCenter with Salesforce to extract data from Salesforce sources and write data to Salesforce targets. Salesforce sources and targets represent objects in the Salesforce object model. Salesforce objects are tables that correspond to tabs and other user interface elements on the Salesforce web site. For example, the Account object contains the information that appears in fields on the Salesforce Account tab. You can view, create, update, and delete data in Salesforce objects.

PowerExchange for Salesforce uses the Salesforce security model to enforce data access controls. Your ability to access data depends on the Salesforce organization, or org, that is associated with the user login you use when you connect to Salesforce. It also depends on the user privileges and the field and row level permissions associated with the login.

You specify the Salesforce login user and password in an application connection object that you create in the Workflow Manager or when you import a Salesforce object in the Designer. PowerExchange for Salesforce uses the Salesforce API to apply existing access policies, and it does not impose additional data access restrictions.

PowerExchange for Salesforce provides the following benefits:

- **Data integration and migration.** Integrate data from diverse data sources, including other applications, with Salesforce data. Also, use PowerCenter to migrate data from various data sources to Salesforce.
- **Data analysis.** Use the built-in data quality capability of PowerCenter to analyze and evaluate Salesforce data.
- **Access to Salesforce functionality through PowerCenter.** Gain full access to Salesforce objects as sources and targets in PowerCenter.

PowerCenter and Salesforce Integration

PowerExchange for Salesforce integrates Salesforce with the Designer so you can import Salesforce objects into PowerCenter and use them in mappings. It integrates Salesforce with the PowerCenter Integration Service so you can run workflows that extract, transform, and load Salesforce data.

PowerExchange for Salesforce uses Transport Layer Security (TLS) to encrypt all data that crosses the Internet. It uses SHA-256 bit encryption. PowerExchange for Salesforce also uses 256-bit encryption before it stores user IDs, passwords, and session IDs in the repository or in temporary storage locations.

Designer and Salesforce Integration

You use the Designer to import Salesforce objects as Salesforce source or target definitions or as Salesforce Lookup, Merge, or Picklist transformations. You can then use those objects in mappings for seamless integration between PowerCenter and Salesforce.

You can import both standard and custom Salesforce objects. Standard object types are objects packaged within Salesforce, such as Account, AccountPartner, Event, Opportunity, and Product.

Custom object types extend the Salesforce data for an organization by defining data entities that are unique to the organization. Salesforce administrators can define custom fields for both standard and custom objects.

The Designer uses a Salesforce login connect to the Salesforce service. It then generates a list of objects that are available for import.

PowerCenter Integration Service and Salesforce Integration

The PowerCenter Integration Service connects to Salesforce to extract, transform, and load Salesforce data. The PowerCenter Integration Service uses a Salesforce login call to authenticate with the Salesforce service. You can specify the login server to use either a production environment or a special testing and development environment called the Salesforce Sandbox.

A connection object stores the Salesforce user ID, password, and end point URL information for the run-time connection. It also stores the security token if you have not added the IP address of the PowerCenter server to the trusted IPs list on Salesforce. Each Salesforce source or target in a mapping references a Salesforce application connection object. You can use multiple Salesforce application connections in a mapping to access different sets of Salesforce data for the sources and targets.

The PowerCenter Integration Service uses the Salesforce security mechanism to authenticate users and manage sessions. The Salesforce API performs user ID and password authentication at the initiation of a run-time session.

At session run time, the PowerCenter Integration Service generates an SOQL query based on the Salesforce source definitions and field projections in the mapping. The SOQL language is a derivative of SQL. The Salesforce API performs SOQL syntax validation at run time.

The PowerCenter Integration Service uses the Salesforce API to read from and write to Salesforce objects. When reading a significant amount of Salesforce data, the PowerCenter Integration Service breaks up the data into smaller segments. PowerExchange for Salesforce submits sequential requests for subsets of a query result set until the entire set has been retrieved.

The PowerCenter Integration Service uses the SOAP protocol to transmit data between the PowerCenter Integration Service and the Salesforce service. To increase performance, the PowerCenter Integration Service uses HTTP compression to reduce the size of the SOAP packets that are sent over the Internet.

When the PowerCenter Integration Service writes data to Salesforce, it converts the PowerCenter datatypes to Salesforce datatypes.

Code Pages

Salesforce processes UTF-8 characters. The PowerCenter Integration Service handles Salesforce data based on the following data movement modes:

- **ASCII.** When the PowerCenter Integration Service runs in ASCII mode, it does not perform any code page validation or any data conversion. You might get inconsistent or truncated data if the PowerCenter Integration Service runs in ASCII mode but processes non-ASCII character data when writing to a Salesforce target.
- **Unicode.** When the PowerCenter Integration Service runs in Unicode mode, it converts data from the source character set to UCS-2, processes the data, and then converts the UCS-2 data to the target code page character set before loading the data. If a session writes to a Salesforce target, the PowerCenter Integration Service converts the UCS-2 data to the UTF-8 character set.

CHAPTER 2

Configuration

This chapter includes the following topics:

Configuration Overview

PowerExchange for Salesforce installs with PowerCenter.

After you install PowerExchange for Salesforce, upgrade it from a previous version, or apply a hotfix, you must register the PowerExchange for Salesforce plug-in with the PowerCenter repository.

Plug-in Registration

After you install PowerExchange for Salesforce, upgrade it from a previous version, or apply a hotfix, it is mandatory to register the PowerExchange for Salesforce plug-in with the PowerCenter repository from the command line interface.

A plug-in is an XML file that defines the functionality of PowerExchange for Salesforce. To register the plug-in, the repository must be running in exclusive mode. Use the *pmrep* RegisterPlugin command to register the plug-in.

The plug-in file for PowerExchange for Salesforce is `pmsfdc.xml`. When you install the Repository component, the installer copies the `pmsfdc.xml` file to the following directory:

```
<Informatica installation directory>/server/bin/native
```

Note: If you do not have the correct privileges to register the plug-in, contact the user who manages the PowerCenter Repository Service.

Registering the Plug-in from the Command Line Interface

To register the Salesforce plug-in by using the pmrep registerplugin command, perform the following steps:

1. Disable the PowerCenter Repository Service.
2. Change the repository mode from **Normal** to **Exclusive**.
3. Enable the PowerCenter Repository Service.
4. On the PowerCenter server machine where repository service is running, navigate to `<$INFA_HOME>/server/bin` and run the following command: `pmrep connect -r <repositoryname> -d <domain_name> -n <Repository_user_name> -x <Repository_password>`
5. Run the following pmrep registerplugin command to register the Salesforce plugin: `pmrep registerplugin -i <$INFA_HOME>/server/bin/native/pmsfdc.xml -e -N`
6. Disable the PowerCenter Repository Service.
7. Change the repository mode from **Exclusive** to **Normal**.
8. Enable the PowerCenter Repository Service.

Salesforce API Version

PowerExchange for Salesforce uses up to version 48.0 of the Salesforce API for source and target transformations. To include Salesforce custom transformations, such as Lookup, Merge, and PickList in a mapping, you must use version 33.0 or below of the Salesforce API.

To use the latest version of the Salesforce API, create an application connection. You can also update the service URL in an application connection. To use version 48.0 of the Salesforce API, use the following Salesforce service URL:

```
https://login.salesforce.com/services/Soap/u/48.0
```

If a Salesforce object has a different structure than a previous version of the object, import the Salesforce object again. After you import the object, analyze associated mappings to determine if you need to update the mappings. Big objects are supported for source and target transformations.

Supported Salesforce Versions

You can use Salesforce API versions that are supported by previous releases of PowerExchange for Salesforce.

The following table lists the Salesforce service URL for each supported Salesforce API:

PowerExchange for Salesforce Version	Salesforce Service URL
10.4.1	https://login.salesforce.com/services/Soap/u/48.0
10.4.0	https://login.salesforce.com/services/Soap/u/47.0 https://login.salesforce.com/services/Soap/u/46.0
10.2.0 HotFix 2	https://login.salesforce.com/services/Soap/u/45.0 https://login.salesforce.com/services/Soap/u/44.0 https://login.salesforce.com/services/Soap/u/43.0

PowerExchange for Salesforce Version	Salesforce Service URL
10.2.0 HotFix 1	https://login.salesforce.com/services/Soap/u/42.0
10.2.0.1.1	https://login.salesforce.com/services/Soap/u/41.0
10.2	https://login.salesforce.com/services/Soap/u/33.0

Java Requirements for Bulk API Target Sessions

When you use the Salesforce Bulk API to write data to Salesforce targets, make sure that the Java temporary directory on the PowerCenter Integration Service machine has at least 10 MB of available space. The recommended amount of space is 50 MB.

When writing data, the Salesforce Bulk API compresses each batch of data to files in the Java temporary directory, and then sends the compressed batch files to Salesforce for processing. Configure at least 10 to 50 MB of space for the Java temporary directory.

You can find the Java temporary directory in the following locations:

- On Windows: `C:\Windows\Temp`.
- On a UNIX node: `/tmp`.

Java Heap Size Configuration

For the Salesforce session to successfully perform the read operation, configure the memory for the Java heap size in the node that runs the PowerCenter Integration Service.

To configure the Java heap size, perform the following steps:

1. In the Administrator Console, navigate to the PowerCenter Integration Service for which you want to change the Java heap size.
2. Click the **Processes** tab.
3. Click **Custom Properties**. The **Edit Custom Properties** dialog box appears.
4. Click **New** to add a new custom property. The following table lists the property names and sample values:
5. Specify the property name and value.

Property Name	Property Value	Sample value
JVMOption1	`-Xmx<memory_size>` to set the maximum heap size	`-Xmx1024m`
JVMOption2	`-Xms<memory_size>` to set the minimum heap size	`-Xms512m`

Note: Specify the maximum and minimum heap size based on the data you want to process.

6. Restart the PowerCenter Integration Service.

HTTP Proxy Options

If you are installing PowerExchange for Salesforce and your organization uses a proxy server to access the internet, you need to configure the HTTP proxy server authentication settings. Configure the HTTP proxy options for the following PowerCenter components:

- PowerCenter Client
- PowerCenter Integration Service

If you are upgrading PowerExchange for Salesforce, verify that the settings for the HTTP proxy options for each component are correct.

Configuring HTTP Proxy Options for the PowerCenter Client

You configure the HTTP proxy options for the PowerCenter Client in a text file named pmsfdc.ini. The file is located in the following directory:

```
<PowerCenter Installation Directory>\clients\PowerCenterClient\client\bin\
```

To configure the HTTP proxy options for the PowerCenter Client:

1. Use a text editor to create a new text file, and save it as "pmsfdc.ini" in the following directory:

   ```
   <PowerCenter Installation Directory>\clients\PowerCenterClient\client\bin\
   ```

   Or, in the <PowerCenter Installation Directory>\clients\PowerCenterClient\client\bin directory, use a text editor to edit the pmsfdc.ini file.
2. Configure the following parameters in the text file and specify appropriate values for each parameter:

Parameter	Required/ Optional	Description
ProxyHost	Required	Proxy host name.
ProxyPort	Required	Proxy port number.
ProxyUser	Required	User name for the proxy account.
ProxyPassword	Required	Password for the proxy account.
Encrypted	Optional	Use when the proxy password is encrypted. The proxy password should be encrypted using the PowerCenter command line program, pmpasswd. Use one of the following options: - DEFAULT. Use for a password encrypted using the CRYPT_DATA encryption type. - SYSTEM. Use for a password encrypted using the CRYPT_SYSTEM encryption type. Required if the proxy password is encrypted.

For example:

```
ProxyHost=d123456.informatica.com
ProxyPort=8082
ProxyUser=admin
ProxyPassword=password123
```

Using an Encrypted Proxy Password

When you configure HTTP proxy options for the PowerCenter Client, you can use an encrypted proxy account password to enhance security.

To use an encrypted password, perform the following tasks:

1. Encrypt the proxy account password using the PowerCenter command line program, pmpasswd.
   You can locate pmpasswd in the following directory on the PowerCenter Integration Service machine:

   ```
   <InformaticaInstallationDir>/server/bin
   ```

   When you encrypt the password, you can use the CRYPT_DATA or CRYPT_SYSTEM encryption type.
2. Use the encrypted password as the proxy password in the pmsfdc.ini file.
3. Add the Encrypted parameter to the pmsfdc.ini file and configure the parameter as follows:
   - Use DEFAULT for passwords encrypted with the CRYPT_DATA encryption type.
   - Use SYSTEM for passwords encrypted with the CRYPT_SYSTEM encryption type.

Configuring HTTP Proxy Options for the PowerCenter Integration Service

If your organization uses a proxy server to access the internet, you must configure the HTTP proxy server authentication settings for PowerCenter Integration Service. To configure the HTTP proxy options for the PowerCenter Integration Service:

1. Open the Administrator tool.
2. Click the **Administration** tab, and then select the PowerCenter Integration Service.
3. Click the **Properties** tab.
4. Click **Edit** in the HTTP Proxy Properties section.
5. Configure the properties described in the following table:

Property	Description
HTTP Proxy Server Host	Name of the HTTP proxy server.
HttpProxyPort	Port number of the HTTP proxy server.
HttpProxyUser	Authenticated user name for the HTTP proxy server.
HttpProxyPassword	Password for the authenticated user.
HttpProxyDomain	Domain for authentication.

CHAPTER 3

Salesforce Sources and Targets

This chapter includes the following topics:

Salesforce Sources and Targets Overview

Use the Designer to import Salesforce source and target definitions into the PowerCenter repository. You can import metadata from any standard or custom Salesforce object available to your Salesforce user account. You can import big objects for source and target transformations.

When you import a Salesforce definition, the Designer creates one port for each field of the Salesforce object. When you import the Salesforce definition, you can also import fields of related objects. The Designer determines related objects based on the relationships defined in Salesforce.

If you create a Salesforce definition manually, the Designer creates the definition with no fields and does not allow you to add any fields.

Importing Fields from Related Salesforce Objects

When you import a Salesforce object, you can import fields of related Salesforce objects into the same source or target definition or transformation. You can import fields from child and descendant related objects.

The Designer identifies related Salesforce objects based on relationships defined in Salesforce. Related objects can include standard and custom Salesforce objects.

For example, if you import Opportunity as a source, you can also import fields of the following child related objects:

- **Account.** By default, the Opportunity object stores the account ID of the opportunity. If you import the related object Account, you can view details about each account, such as account name, account type, and the parent account.

- **Campaign.** By default, the Opportunity object stores the campaign ID of the opportunity. If you import the related object Campaign, you can view details about the corresponding campaign, such as campaign name, type, description, and start and end dates.

In this example, the Opportunity object is the primary Salesforce object in the definition and the Account and Campaign objects are child related objects. A primary Salesforce object is the primary object used to create the definition.

You can also import descendant related objects, which are indirectly related to the primary object. For example, when you import the Opportunity object, you can also import the User descendant related object based on its Accounts relationship with the Account child related object. You might import the fields of the User object to get the details about the account owner such as the account owner name, phone number, and email address.

When you import the primary Salesforce object in the Designer, you can also specify the related objects to import. The Designer displays related objects using the following naming convention:

```
<Relationship name> (<Salesforce object name>)
```

The Designer shows the relationship name and the object name as they are defined in Salesforce.

Note: When you import a Salesforce object and its related objects, some related objects may not appear. The Salesforce API does not expose all related objects and their relationships.

Rules and Guidelines for Importing Fields from Related Salesforce Objects

Use the following rules and guidelines when you import fields from related Salesforce objects:

- Import fields of related objects to create a Salesforce source definition under the following circumstances:
  - You want to extract data from Salesforce only.
  - You want to load data for the Salesforce object and one or more related objects.
  - You do not need to transform source data before you load it into the target.
  - To improve session performance when you join data from multiple Salesforce objects. Instead of using a Joiner transformation to join data from multiple Salesforce objects, import the fields of the related objects in the Salesforce source definition to improve performance.
- You cannot import fields from a related object when the related object is a parent object.
- You cannot import fields from a related object when the related object is not queryable, such as the Attachment, Note, or OpenActivity objects. For information about objects that are not queryable, see the Salesforce documentation.
- Import fields of related objects to create a Salesforce target definition when you want to upsert records into a Salesforce target and its related objects. You can use the external ID or Salesforce idLookup fields to upsert records into a Salesforce target or related object.
- Import fields of related objects for a Salesforce Lookup transformation to look up data in the primary Salesforce object and related objects without performing a join. For example, you can look up opportunity information in the Opportunity object and also get the related account information in the Account related object.
- Import fields of related objects for a Salesforce Merge transformation when you want to merge data in the primary Salesforce object with data in related objects.
- Import fields of related objects for a Salesforce Picklist transformation when you want to retrieve picklist values from related objects.

Salesforce Sources

The PowerCenter Integration Service can extract data from Salesforce source definitions. You can import one or more source definitions for each Salesforce object. If you import multiple source definitions for the same Salesforce object, you must provide unique names for the source definitions. If you do not provide unique names, the Designer requires you to resolve the conflict.

When the PowerCenter Integration Service extracts data from a Salesforce source, it converts the data based on the datatypes in the Application Source Qualifier associated with the source.

Time Zones for Salesforce Sources

Salesforce can store datetime data in multiple time zones. The PowerCenter Integration Service converts the time zones of all times extracted from Salesforce to the Coordinated Universal Time (UTC) time zone.

Time Conversion from Salesforce Sources

When the PowerCenter Integration Service extracts data from a Time field of a Salesforce object, it adds the current date to the time. For example, the PowerCenter Integration Service extracts 10:46:51.9884 from a Time field in a Salesforce source on 01/01/2007. The resulting Date/Time value is 01/01/2007 10:46:51.988400.

Salesforce Targets

You can perform insert, update, delete, and upsert operations on a Salesforce target.

The PowerCenter Integration Service determines whether records are new, existing, or deleted based on the record ID. You can use the following types of IDs to identify records in Salesforce objects:

- Salesforce ID. By default, Salesforce generates IDs for each new record.
- External ID. External IDs are IDs that are generated outside of Salesforce. You cannot use external IDs to identify records for update or deletion from a Salesforce target.

You can also configure the PowerCenter Integration Service to replace existing values in a record in a Salesforce target with null values during an update or upsert operation. By default, the PowerCenter Integration Service does not replace existing values with null values.

Related Topics:

- "Handling Null Values in Update and Upsert Operations" on page 47

Time Zones for Salesforce Targets

Salesforce targets store time data as Coordinated Universal Time (UTC) data. Ensure that each mapping converts time data to the UTC time zone before it loads the data into the target.

Determining Possible Update Strategies for a Salesforce Target

You can view properties for a Salesforce target definition to determine whether you can create, update, or upsert records into the Salesforce target. To view details about object fields, double-click the title bar of the target definition and select the Attributes tab.

The Attributes tab lists attribute values for each field in the imported Salesforce object.

The following table describes the attributes that display for each field:

Attribute	Description
SforceName	Field name in Salesforce.
Field Name	Name of the field as defined in the Salesforce target definition.
Createable	Indicates if you can insert data in the field: - 0. You cannot insert data in this field. - 1. You can insert data in this field. A PowerCenter insert is equivalent to a Salesforce create operation.
Updateable	Indicates if you can update data in the field: - 0. You cannot update data in this field. - 1. You can update data in this field.
External ID	Salesforce custom fields only. Indicates if the field is an external ID field: - 0. The field is not an external ID field. - 1. The field is external ID field. Each Salesforce object can include a single custom field designated as the external ID field. Salesforce appends custom field names with "__c".
idLookup	Indicates if the field is an idLookup field: - 0. The field is not an idLookup field. - 1. The field is an idLookup field. You can use an idLookup field to perform upserts on a Salesforce standard object.

For more information about Salesforce attributes, see the Salesforce documentation.

Rules and Guidelines for the Salesforce Target Update Strategy

Use the following rules and guidelines when configuring the update strategy for a Salesforce target:

- The PowerCenter Integration Service cannot create or update system fields in a record of a Salesforce target. Each Salesforce object includes read-only system fields that Salesforce creates or updates. For example, each Salesforce object includes an Id system field. The Id field contains a unique identifier for the record. When you run a session containing a Salesforce target definition, the PowerCenter Integration Service inserts the record into the Salesforce target, but Salesforce generates the ID.
- To configure a session to update or delete records in the Salesforce target, you must pass the ID for each record through the mapping and link it to the Id input port in the Salesforce target definition. Salesforce uses the Id field to identify the record being updated or deleted.
- To upsert a record, you must provide the external ID or Salesforce idLookup field.
- To delete a record, you must provide the Salesforce ID.
- You do not need to link the Id input port for a session that inserts records. Salesforce generates the ID for new records. You can use the upsert operation to insert and update records based on the external ID if you use the external ID field to identify records in a Salesforce object. You might choose to use an external ID to update or upsert records when it is difficult to get the ID from the Id field in the Salesforce target.
- If the PowerCenter Integration Service encounters a transient error while performing an update, upsert, or delete operation, it retries the operation. The PowerCenter Integration Service retries the operation three

times with a five-minute interval between retry attempts. Transient errors include errors such as network failures and timeouts from Salesforce.

- Salesforce targets cannot contain XML characters. If the PowerCenter Integration Service encounters XML characters in a source field, it removes the XML characters before loading the Salesforce target field.

Importing a Salesforce Source or Target Definition

To import a source or target definition from Salesforce:

1. Import a definition.
   - To import a Salesforce source definition, in the **Source Analyzer**, click **Sources** > **Import from Salesforce**.
   - To import a Salesforce target definition, in the **Target Designer**, click **Targets** > **Import from Salesforce**.
2. In the **Import from Salesforce** dialog box, you can select the connection type as **Standard** or **OAuth**.

   The following table lists the properties for a standard connection:

Connection Property	Description
Connection Type	Select Standard.
Username	Username for the Salesforce account.
Password	Password for the Salesforce account.
Service URL	URL of the Salesforce service. `https://login.salesforce.com/services/Soap/u/48.0`

   The following table lists the properties for an OAuth connection:

Connection Property	Description
Connection Type	Select OAuth.
Refresh Token	The refresh token of Salesforce. For more information about how to generate the refresh token, see "Generating the Refresh Token" on page 38
Consumer Key	The consumer key obtained from Salesforce, required to generate a valid refresh token. For more information about how to generate the consumer key, see the Salesforce documentation.
Consumer Secret	The consumer secret obtained from Salesforce, required to generate a valid refresh token. For more information about how to generate the consumer secret, see the Salesforce documentation.

3. Click **Connect**.

The Designer displays a list of objects available to the Salesforce user.

4. Select the objects you want to import, and click **OK**.
   - Hold down the Shift key to select blocks of tables.
   - Hold down the Ctrl key to make non-contiguous selections.
   - Use the Select All button to select all tables.
   - Use the Select None button to clear all highlighted selections.
5. In the Include Salesforce Object Relationship dialog box, select the related objects to include in the source definition.
6. Click **OK**.

CHAPTER 4

Salesforce Lookup Transformation

This chapter includes the following topics:

Salesforce Lookup Transformation Overview

The Salesforce Lookup transformation is an active transformation. Use a Salesforce Lookup transformation to look up data in a Salesforce object. For example, the source table includes the employee ID, but you want to include the employee name in the target table to make summary data easy to read. You can use the Salesforce Lookup transformation to look up the employee name in a Salesforce object. Use version 33.0 or below of the Salesforce API when you include a Salesforce Lookup transformation in a mapping.

You can create a Salesforce Lookup transformation from any standard or custom Salesforce object available to your Salesforce user account. You can also include fields from related Salesforce objects in the Salesforce Lookup transformation.

The PowerCenter Integration Service queries the lookup source based on ports in the transformation. It generates queries in sforce Object Query Language (SOQL), which is a derivative of SQL. It generates a query for each row that enters the Salesforce Lookup transformation. The PowerCenter Integration Service compares the transformation port values to lookup source field values based on the SOQL queries.

A Salesforce Lookup transformation differs from a Lookup transformation. For example, the Salesforce Lookup transformation returns all rows that match the lookup query condition. The Lookup transformation returns one row. When the Salesforce Lookup transformation returns multiple rows, it assigns a sequence ID to each row that matches the condition.

The following table describes the differences between the Salesforce Lookup transformation and the Lookup transformation:

Item	Salesforce Lookup Transformation	Lookup Transformation
Lookup object	Salesforce object	Flat file or relational table
Transformation type	Active	Passive
Transformation in mapping	Connected	Connected or unconnected
Cache type	Uncached	Cached or uncached
Return rows	All matched rows	Single matched row
Query language	SOQL query	SQL query
Query condition configuration	Based on connected ports	Defined on the Condition tab of the transformation

Note: The Salesforce Lookup transformation can retrieve data in real time. To cache Salesforce data, use a Salesforce source definition instead of a Salesforce Lookup transformation. You can use a Joiner transformation to join two data sources and cache source data during processing. The Data Integration Service ignores conditions that you enter in the Join query.

Salesforce Lookup Components

When you configure a Salesforce Lookup transformation, you define the following components:

- **Transformation tab.** You can rename the transformation and add a description on the Transformation tab.
- **Ports tab.** The Salesforce Lookup transformation can include lookup ports, pass-through ports, and default ports.
- **Properties tab.** You can configure the tracing level for the transformation. The default tracing level is Normal. In addition, you can specify whether or not the output is deterministic. By default, the output is deterministic.
- **Initialization Properties tab.** The Salesforce Lookup transformation does not use initialization properties. The PowerCenter Integration Service retrieves initialization information from a vendor-defined metadata extension.
- **Metadata Extensions tab.** Create a non-reusable metadata extension to extend the metadata of the transformation transformation. Configure the extension name, datatype, precision, and value. You can also promote a metadata extension to be reusable if you want to make it available to all transformations.
- **Port Attribute Definitions tab.** The Port Attribute Definitions tab displays the port attributes defined for a Salesforce Lookup transformation. SforceDataType and SforceName are the only port attributes.

The transformation includes some configurable Custom transformation properties that the PowerCenter Integration Service does not use for lookups. The PowerCenter Integration Service ignores those configurable properties at run time.

The following table lists configurable properties that the Salesforce Lookup transformation does not use:

Property	Location	Description
Runtime Location	Properties tab	The PowerCenter Integration Service ignores the run-time location.
Initialization Property	Initialization Properties tab	The PowerCenter Integration Service ignores the initialization properties. It retrieves initialization information from a vendor-defined metadata extension.
Port Attribute Definition	Port Attribute Definitions tab	The Salesforce Lookup transformation uses a port attribute definition named SforceDataType. If you configure additional port attribute definitions, the PowerCenter Integration Service ignores them.

Salesforce Lookup Ports

You can include the following types of ports in a Salesforce Lookup transformation:

- **Lookup ports.** You import the lookup ports from a Salesforce object definition when you create a Salesforce Lookup transformation. The PowerCenter Integration Service uses lookup ports to generate the lookup query.
- **Pass-through ports.** You can add pass-through ports to the transformation. The PowerCenter Integration Service passes these port values unchanged through the transformation.
- **Default ports.** When you import a Salesforce object definition, the Designer creates default ports named LKP_FILTER and LKP_MATCHIDX. You can use the LKP_FILTER port to add a filter condition to the lookup query. The PowerCenter Integration Service uses the LKP_MATCHIDX port to assign sequence IDs to matched rows.

Lookup Ports

When you import a Salesforce object to create a Salesforce Lookup transformation, the Designer creates ports to use in the lookup query. The PowerCenter Integration Service generates the lookup query based on connected input and output ports. If you rename a port that you imported from a Salesforce object, the PowerCenter Integration Service does not include the port in the lookup query.

When you import a Salesforce object, the Designer converts the Salesforce field datatypes to transformation datatypes and stores the Salesforce datatypes as port attributes. The Ports tab of a Salesforce Lookup transformation displays the transformation datatypes. To view the Salesforce datatypes, view the port-level attributes.

To view the port-level attributes in a Salesforce Lookup transformation, click the Ports tab of the transformation. Then click Edit > Port Attributes. The Salesforce datatypes for ports appear in the port-level attributes.

For ports that are imported from Salesforce, the datatypes must conform to the Salesforce and transformation datatype mapping.

Related Topics:

- "Salesforce Lookup Query" on page 27
- "Salesforce and Transformation Datatypes" on page 54

Pass-Through Ports

You can add ports to a Salesforce Lookup transformation as pass-through ports. The PowerCenter Integration Service passes the value of these ports through the transformation without performing lookup on the data.

To add a pass-through port, create a new port or copy and paste a port into the transformation. Then, connect the port to upstream and downstream ports.

LKP_FILTER Port

When you import a Salesforce object to create a Salesforce Lookup transformation, the Designer creates a default port named LKP_FILTER. Use the LKP_FILTER port to add filter conditions in the lookup query that you cannot generate by connecting the lookup input ports of the Lookup transformation.

Note: You can use the LKP_FILTER port in conjunction with the connected lookup input ports.

In a mapping, you can create a transformation such as an Expression transformation that outputs a constant, a range, or a mapping parameter or variable value. Then, you can connect the appropriate output port of the transformation to the LKP_FILTER input port. The lookup query includes the output in the WHERE clause of the lookup query.

For example, you create an Expression transformation that outputs the value of the Name port as a constant, 'Edge Communications.' The transformation uses the following expression:

```
'Name ='  || CHR(39) || 'Edge Communications'  || CHR(39)
```

Then, you project the Expression transformation output for the Name port to the LKP_FILTER port in the Salesforce Lookup transformation. In this example, the Salesforce Lookup transformation is based on the Salesforce object named Account. The connected lookup input ports are Id, Sale_Amount, and Sale_Date. The connected lookup output ports are Sale_Amount, Sale_Date, and Name.

Note: If you use a default port and modify or rename it, you might get unexpected results.

The SOQL SELECT statement for each row that passes through the transformation has the following format:

```
SELECT Id, Name, Phone FROM Account WHERE Id = '<value of Id>' AND AccountNumber =
'<value of AccountNumber>' AND Name = 'Edge Communications'
```

Related Topics:

- "Salesforce Lookup Query" on page 27

LKP_MATCHIDX Port

When you import a Salesforce object to create a Salesforce Lookup transformation, the Designer creates a default port named LKP_MATCHIDX.

Use the LKP_MATCHIDX output port to identify the matched rows when a lookup query returns multiple matches. For each matched row, the PowerCenter Integration Service assigns a unique sequence ID. A value of 0 means that no match exists for the input row.

The following rules apply to matched and unmatched rows:

- **Matched.** The SOQL query returns a row for each match found. It can return multiple rows for one input row. If you use the LKP_MATCHIDX port and the query returns multiple matches, the PowerCenter Integration Service generates a sequence ID for each returned row. The values of the sequence IDs are 1–*n*.
- **Unmatched.** If the SOQL query returns no rows, the PowerCenter Integration Service generates one output row with a sequence ID of 0. The PowerCenter Integration Service retains the value of pass-through ports, and it sets the value of the lookup output ports to NULL. To pass null values to the target, you must configure it to accept null values. If you do not want to pass null values to the target, use a transformation to convert null values to a default value.

Note: If you use a default port and modify or rename it, you might get unexpected results.

Salesforce Lookup Query

The input and output ports in a Salesforce Lookup transformation determine the sforce Object Query Language (SOQL) query used in the lookup. The SOQL language is a derivative of SQL. The PowerCenter Integration Service generates a separate SOQL query for each row that passes into the transformation. The port values for each row determine the values used in the query.

The PowerCenter Integration Service generates SOQL queries according to the following rules:

- A lookup port is a port that you import from a Salesforce object and connect in a mapping.
- All lookup ports must match the Salesforce field name and have an associated Salesforce datatype.
- The PowerCenter Integration Service generates the SELECT statement based on the connected lookup output ports.
- The PowerCenter Integration Service generates the WHERE clause based on the connected lookup input ports and the LKP_FILTER port.
- The PowerCenter Integration Service matches rows based on equality with the connected input port values and, if applicable, based on an additional filter condition.

For example, a Salesforce Lookup transformation is based on the Salesforce object named Account. The connected lookup input ports are Id and AccountNumber. The connected lookup output ports are Id, Name, and Phone. The SOQL SELECT statement has the following format:

```
SELECT Id, Name, Phone FROM Account WHERE Id = '<value of Id>' AND AccountNumber =
'<value of AccountNumber>'
```

Note: When you use the SOQL query with the Filter transformation or Expression transformation, the Data Integration Service ignores the SOQL query.

Creating a Salesforce Lookup Transformation

To create a Salesforce Lookup transformation:

1. In the Transformation Developer or Mapping Designer, click Transformation > Create.

   The Create Transformation dialog box appears.
2. Select Salesforce Lookup as the transformation type, and enter a name.

3. Click Create.
4. In the Import Tables from Salesforce dialog box, enter the following information:

Import Attribute	Description
User Name	Salesforce user name.
Password	Password for Salesforce user name. The password is case sensitive.
Service URL	URL of the Salesforce service. `https://login.salesforce.com/services/Soap/u/33.0` In a test or development environment, you might want to access the Salesforce Sandbox testing environment. For more information about the Salesforce Sandbox, see the Salesforce documentation.

5. Click Connect.

   The Designer displays a list of objects available to the Salesforce user.
6. Select the object you want to import, and click OK.
7. In the Include Salesforce Object Relationships dialog box, select the relationships to include, and click OK.

   If there are no conflicts, the Designer creates a Salesforce Lookup transformation based on the selected object and related objects. If there are conflicts, resolve the conflicts.
8. In the Create Transformation dialog box, click Done.

CHAPTER 5

Salesforce Merge Transformation

This chapter includes the following topics:

Salesforce Merge Transformation Overview

The Salesforce Merge transformation is a passive transformation. Use a Salesforce Merge transformation to merge duplicate records.

You can create a Salesforce Merge transformation from Account, Contact or Lead Salesforce objects. You can also include fields from related Salesforce objects in the Salesforce Merge transformation.

You can merge up to three Salesforce records into a single record. When you create a Salesforce Merge transformation, you identify the master record and up to two slave records. When you merge the records, the Salesforce Merge transformation retains the master record and deletes the slave records from the Salesforce object.

To ensure that no child records become orphaned, the Salesforce Merge transformation reassigns child records of slave records to the master record. For example, you merge two records from the Account Salesforce object. Each account record is the parent of a record in the Contact Salesforce object. When the Salesforce Merge transformation merges the account records, it also assigns the contact record associated with the slave account record to the master account record.

You can override the values of Salesforce object attributes in a master record. To override an attribute value, configure the source to provide the value for the Salesforce object attribute, and map the field from the source through the Salesforce Merge transformation to the target in the mapping.

Sample Salesforce Merge Transformation

You want to merge the following records in the Account Salesforce object:

ID	Account Name	Billing City	Phone
0015000000IcEgAAAV	ABC Tiles	Los Angeles	NULL
0015000000IcEgBAAV	ABC Tiles	NULL	310-555-1212
0015000000IcEgCAAV	ABC	San Diego	310-555-6666

The following table shows the source for the mapping that merges the records and updates the account name in the Account object:

Master Record ID	Slave 1 Record ID	Slave 2 Record ID	Account Name
0015000000IcEgAAAV	0015000000IcEgBAAV	0015000000IcEgCAAV	Textiles to Go

The account name changed from ABC Tiles to Textiles to Go. You project the Textiles to Go value for the Account Name field to the Salesforce Merge transformation to the target.

When you run the session that contains the mapping, the PowerCenter Integration Service deletes the slave records from the Account Salesforce object and retains the following master record:

ID	Account Name	Billing City	Phone
0015000000IcEgAAAV	Textiles to Go	Los Angeles	NULL

The Salesforce Merge transformation does not overwrite the Account object attributes in the master record with the values from the slave records. It does update the Account Name attribute because the value is provided.

Salesforce Merge Components

When you configure a Salesforce Merge transformation, you define the following components:

- **Transformation tab.** You can rename the transformation and add a description on the Transformation tab.
- **Ports tab.** The Salesforce Merge transformation can include Salesforce object attribute ports and default ports.
- **Properties tab.** You can configure the tracing level for the transformation. The default tracing level is Normal. In addition, you can specify whether or not the output is deterministic. By default, the output is deterministic.
- **Initialization Properties tab.** The Salesforce Merge transformation does not use initialization properties. The PowerCenter Integration Service retrieves initialization information from a vendor-defined metadata extension.
- **Metadata Extensions tab.** Create a non-reusable metadata extension to extend the metadata of the transformation transformation. Configure the extension name, datatype, precision, and value. You can

also promote a metadata extension to a reusable extension if you want to make the extension available to all transformation transformations.

- **Port Attribute Definitions tab.** The Port Attribute Definitions tab displays the port attributes defined for a Salesforce Merge transformation. SforceDataType and SforceName are the only port attributes.

Salesforce Merge Ports

You can include the following types of ports in a Salesforce Merge transformation:

- **Salesforce object attribute ports.** Use these ports to update values for the Salesforce object attributes in the master record. The values in these ports override the values in the master and slave records.
- **Default ports.** When you import a Salesforce object definition, the Designer creates the default input ports and the default MergedID, MergedSlaveID1, and MergedSlaveID2 output ports.

Salesforce Object Attribute Ports

The Salesforce Merge transformation contains input ports for all attributes of the Salesforce object. Provide values for each port to override the values in the master record. The PowerCenter Integration Service updates the master record with the values provided for each Salesforce object port.

When you map source values to a port, verify that the datatypes the source data and ports are compatible. If you map incompatible datatypes, the session may fail or the master record may contain unexpected values.

Related Topics:

- "Salesforce and Transformation Datatypes" on page 54

ID and SlaveID Input Ports

Map the master record ID to the ID input field. Map the slave record ID to the input SlaveID1 field. If applicable, map the second slave record ID to the input SlaveID2 field. The PowerCenter Integration Service identifies the master and slave records in the Salesforce object using the IDs provided by the source.

MergedID, MergedSlaveID1, and MergedSlaveID2 Output Ports

The Salesforce Merge transformation populates the master record ID in the MergedID field. The PowerCenter Integration Service also deletes slave records that are identified in the SlaveID1 and SlaveID2 fields.

Rules and Guidelines for the Salesforce Merge Transformation

Use the following rules and guidelines to configure the Salesforce Merge transformation:

- Use version 33.0 or below of the Salesforce API when you include a Salesforce Merge transformation in a mapping.
- The session fails if you do not specify the master record ID and at least one slave ID.

- The IDs for the master record and slave records must be valid Salesforce IDs.
- The values of fields that are linked to the Salesforce Merge transformation overwrite existing values in the master record. However, null values in linked fields do not overwrite valid values in the master record.
- The Salesforce Merge transformation reassigns child records of each slave record to the master record during the merge.
- You can merge at most two slave records with a master record. To merge more than two slave records, create and run the Salesforce Merge transformation multiple times using the same master record ID.
- The PowerCenter Integration Service does not merge a slave record if the slave ID is blank, does not exist in the Salesforce object, or the slave ID does not contain 18 characters.
- If the master ID or at least one slave ID does not exist in the Salesforce object, the merge fails.

Creating a Salesforce Merge Transformation

To create a Salesforce Merge transformation:

1. In the Transformation Developer or Mapping Designer, click Transformation > Create.

   The Create Transformation dialog box appears.
2. Select Salesforce Merge as the transformation type, and enter a name.
3. Click Create.
4. In the Import Tables from Salesforce dialog box, enter the following information:

Import Attribute	Description
User Name	Salesforce user name.
Password	Password for Salesforce user name. The password is case sensitive.
Service URL	URL of the Salesforce service. `https://login.salesforce.com/services/Soap/u/33.0` In a test or development environment, you might want to access the Salesforce Sandbox testing environment. For more information about the Salesforce Sandbox, see the Salesforce documentation.

5. Click Connect.

   The Designer displays a list of objects available to the Salesforce user.
6. Select the object you want to import, and click OK.
7. In the Include Salesforce Object Relationships dialog box, select the relationships you want to include, and click OK.

   If there are no conflicts, the Designer creates a Salesforce Merge transformation based on the selected object and related objects. If there are conflicts, resolve the conflicts.
8. In the Create Transformation dialog box, click Done.

CHAPTER 6

Salesforce PickList Transformation

This chapter includes the following topics:

Salesforce PickList Transformation Overview

The Salesforce PickList transformation is a passive transformation. Use the Salesforce PickList transformation to retrieve a list of picklist values for a field in a Salesforce object. A Salesforce picklist is a list of valid values for a Salesforce field. You might retrieve picklist values to validate source data.

You can create a Salesforce PickList transformation from any standard or custom Salesforce object available to your Salesforce user account. You can also include fields from related Salesforce objects in the Salesforce PickList transformation.

The PowerCenter Integration Service can retrieve the picklist for each field with a defined picklist in the Salesforce PickList transformation. In the mapping, you can use any source type with the Salesforce PickList transformation. To retrieve the list of picklist values, connect the Source Qualifier transformation to the PickList_Input port of the Salesforce PickList transformation. The transformation outputs a colon-separated list of valid values for the picklist fields linked to the target.

Salesforce PickList Components

When you configure a Salesforce PickList transformation, you define the following components:

- **Transformation tab.** You can rename the transformation and add a description on the Transformation tab.
- **Ports tab.** The Salesforce PickList transformation can include picklist ports and pass-through ports.

- **Properties tab.** You can configure the tracing level for the transformation. The default tracing level is Normal. In addition, you can specify whether or not the output is deterministic. By default, the output is deterministic.
- **Initialization Properties tab.** The Salesforce PickList transformation does not use initialization properties. The PowerCenter Integration Service retrieves initialization information from a vendor-defined metadata extension.
- **Metadata Extensions tab.** Create a non-reusable metadata extension to extend the metadata of the transformation transformation. Configure the extension name, datatype, precision, and value. You can also promote a metadata extension to a reusable extension if you want to make the extension available to all transformation transformations.
- **Port Attribute Definitions tab.** The Port Attribute Definitions tab displays the port attributes defined for a Salesforce PickList transformation. The SforceDataType and SforceName are the only port attributes.

Salesforce PickList Ports

You can include the following types of ports in a Salesforce PickList transformation:

- **PickList_Input port.** To enable the PowerCenter Integration Service to retrieve the Salesforce picklist, map any field in the Source Qualifier transformation to the PickList_Input port in the Salesforce PickList transformation. If you do not map a field to the PickList_Input port, the session fails.
- **Output ports.** By default, the Designer creates output ports for Salesforce object attributes that have picklist values. To output multiple picklists for different Salesforce object attributes, configure one Source Qualifier transformation for each attribute. When you run the session, the PowerCenter Integration Service outputs picklist values for each Salesforce object attribute included in the mapping.
- **Pass-through ports.** You can add ports to a Salesforce PickList transformation as pass-through ports. The PowerCenter Integration Service passes the value of a pass-through port through the transformation and into the target if the port is connected to the target.

Rules and Guidelines for the Salesforce PickList Transformation

Use the following guidelines to configure the Salesforce PickList transformation:

- Use version 33.0 or below of the Salesforce API when you include a Salesforce PickList transformation in a mapping.
- The default length for picklist ports is 512 characters. The PowerCenter Integration Service truncates values that are longer than 512 characters. To prevent truncation, increase the port length.
- You must connect at least one port from the source to the Salesforce PickList transformation.
- The Salesforce PickList transformation does not retrieve restricted picklist values from Salesforce because the Salesforce API does not expose them.
- The Salesforce PickList transformation does not retrieve values from multi-select picklists.

Creating a Salesforce PickList Transformation

To create a Salesforce PickList transformation:

1. In the Transformation Developer or Mapping Designer, click Transformation > Create.

   The Create Transformation dialog box appears.
2. Select Salesforce PickList as the transformation type, and enter a name.
3. Click Create.
4. In the Import Tables from Salesforce dialog box, enter the following information:

Import Attribute	Description
User Name	Salesforce user name.
Password	Password for Salesforce user name. The password is case sensitive.
Service URL	URL of the Salesforce service. `https://login.salesforce.com/services/Soap/u/33.0` In a test or development environment, you might want to access the Salesforce Sandbox testing environment. For more information about the Salesforce Sandbox, see the Salesforce documentation.

5. Click Connect.

   The Designer displays a list of objects available to the Salesforce user.
6. Select the object you want to import, and click OK.
7. In the Include Salesforce Object Relationships dialog box, select the relationships to include, and click OK.

   If there are no conflicts, the Designer creates a Salesforce PickList transformation based on the selected object and related objects. If there are conflicts, resolve the conflicts.
8. In the Create Transformation dialog box, click Done.

CHAPTER 7

Salesforce Sessions and Workflows

This chapter includes the following topics:

Salesforce Sessions and Workflows Overview

After you create mappings, you can create a session and use the session in a workflow to extract, transform, and load data. Create sessions and workflows in the Workflow Manager.

When you configure a Salesforce session, you create connections to read data from and write data to Salesforce. You can also define properties in a session to determine how the PowerCenter Integration Service reads data from a Salesforce source or writes data to a Salesforce target.

To configure the session, complete the following tasks:

- Configure an application connection for Salesforce sources and targets in the Workflow Manager. You configure application connections to read from or write to Salesforce.
- Configure the session properties for the Salesforce source.
- Configure the session properties for the Salesforce target.
- Optionally, configure the session for optimal performance.

Salesforce Connections

Before the PowerCenter Integration Service can connect to Salesforce, you must configure a Salesforce application connection in the Workflow Manager.

When you configure a Salesforce application connection, you specify connection attributes the PowerCenter Integration Service uses to connect to Salesforce. A connection object stores the Salesforce user ID, password, and end point URL information for the runtime connection.

You can use one of the following authentication types to connect to Salesforce:

- **Standard**. You must provide the user name and password to connect to Salesforce.
- **OAuth**. OAuth allows secure API authorization. You do not need to disclose your Salesforce credentials and the Salesforce administrator can revoke the access at any time. You must provide the service URL, refresh token, consumer key, and consumer secret to connect to Salesforce.

Configuring a Salesforce Connection

To configure a Salesforce application connection:

1. In the **Workflow Manager**, connect to a PowerCenter repository.
2. Click **Connections** > **Application**.

   The **Application Connection Browser** dialog box appears.
3. From **Select Type**, select Salesforce Connection.
4. Click **New**.

   The **Connection Object Definition** dialog box appears.
5. Select the connection type as **Standard** or **OAuth**.

   The connection properties that you configure differ based on the authentication type that you select.

   The following table lists the properties for a standard connection:

Connection Property	Description
Name	Name of the Salesforce connection.
Type	Standard Salesforce connection.
User Name	User name for the Salesforce account.
Password	Password for the Salesforce account.

Service URL	Enter the following URL of the Salesforce service:
	`https://login.salesforce.com/services/Soap/u/48.0`
	Enter the following URL of the Salesforce service while using Salesforce Picklist, Merge, and Lookup transformation:
	`https://login.salesforce.com/services/Soap/u/33.0`

   The following table lists the properties for an OAuth connection:

Connection Property	Description
Name	Name of the OAuth connection.
Type	Select the **Use OAuth** checkbox to use the OAuth connection.

Connection Property	Description
Refresh Token	The refresh token used to get a fresh access token after it expires. For more information about how to generate the Refresh Token, see "Generating the Refresh Token" on page 38
Consumer Key	The consumer key obtained from Salesforce required to generate the refresh token. For more information about how to generate the consumer key, see the *Salesforce documentation*.
Consumer Secret	The consumer secret obtained from Salesforce required to generate the refresh token. For more information about how to generate the consumer secret, see the *Salesforce documentation*.

6. Click **OK**.

   The new application connection appears in the Application Object Browser.

Generating the Refresh Token

The SFDC OAuth Tool generates the refresh token using the consumer key and consumer secret. Use the refresh token to obtain a fresh access token after it expires.

You must create the Salesforce Connected App to get the consumer key and consumer secret. Perform the following steps to generate the refresh token:

1. Download the SFDC OAuth Tool from the following link:

   https://marketplace.informatica.com/listings/cloud/solutions/sfdc_oauth_tool_for_powercenter_10_2_0_hotfix_1.html
2. Extract the `OAuth.zip` file.
3. Go to the `oauth\conf` folder, open the `server.xml` file, and update the `mystore.jks` file path to the file path on your system.
4. Save and close the file.
5. Go to `~\oauth\bin` and run the command `catalina.bat start`.
6. Go to `https://localhost:8090/salesforce` from a browser.
7. Enter your Salesforce user name and password to log in.
8. Enter the **Client Id** and **Client Secret Key**.

   The Client Id is the consumer key and Client Secret Key is the consumer secret that you get from the Salesforce Connected App.
9. Click **Submit** to generate the refresh token.

Troubleshooting a Salesforce Connection

Consider the following troubleshooting tip when you create a Salesforce connection:

Invalid login error

The PowerCenter Integration Service uses the Salesforce security mechanism to authenticate the login. If you specify the login details that are not valid, the following error appears:

```
Invalid login. When accessing Salesforce from outside of your company's trusted
networks, you must append a security token to your password to log in to the API or
```

```
a desktop client. To receive or reset your security token, log in to Salesforce with
your browser and click Setup | My Personal Information | Reset Security Token.
```

For more information about getting a valid Salesforce login, contact the Salesforce administrator for your organization.

Configuring a Session with a Salesforce Source

You can configure the session properties for a Salesforce source on the Mapping tab. Define the properties for each source instance in the session.

The following table describes the session properties you can configure for a Salesforce source:

Property Name	Description
SOQL Filter Condition	Enter a filter condition to filter Salesforce source records.
CDC Time Limit	Time period, in seconds, that the PowerCenter Integration Service reads changed Salesforce data. When you set the CDC Time Limit to a non-zero value, the PowerCenter Integration Service performs a full initial read of the source data and then captures changes to the Salesforce data for the time period you specify. Set the value to -1 to capture changed data for an infinite period of time. Default is 0.
Flush Interval	Interval, in seconds, at which the PowerCenter Integration Service captures changed Salesforce data. Default is 300. If you set the CDC Time Limit to a non-zero value, the PowerCenter Integration Service captures changed data from the source every 300 seconds. Otherwise, the PowerCenter Integration Service ignores this value. The minimum recommended value for Flush Interval is 60 seconds.
CDC Start Timestamp	Start date and time for the time period. The PowerCenter Integration Service extracts data that was added or modified after this time. Must be in the format YYYY-MM-DDTHH:MI:SS.SSSZ. You can also use the $Paramstart mapping variable in a parameter file to specify the CDC start time.
CDC End Timestamp	End date and time for the time period. The PowerCenter Integration Service extracts data that was added or modified before this time. Must be in the format YYYY-MM-DDTHH:MI:SS.SSSZ. You can also use the $Paramend mapping variable in a parameter file to specify the CDC end time.
Row Limit	The maximum number of rows the PowerCenter Integration Service processes. Default is 0. 0 indicates that there is no row limit, and the PowerCenter Integration Service processes all records.
Use queryAll	Runs a query that returns all rows, including active, archived, and deleted rows that are available in the recycle bin. Otherwise, the PowerCenter Integration Service returns only active rows. The PowerCenter Integration Service ignores this property when you configure the session to use the Enable Bulk Query option or to perform change data capture.
Use SystemModstamp for CDC	Uses the SystemModstamp as the timestamp for changed records in Salesforce. Otherwise, the PowerCenter Integration Service uses the LastModifiedDate timestamp to identify changed records in Salesforce. Default is to use the LastModifiedDate timestamp.

Property Name	Description
CDC Flush Interval Offset	The number of seconds that you want to offset the CDC flush interval. Captures real-time data that is submitted within the CDC time limit but not committed by Salesforce within the time limit.
Enable Bulk Query	Uses the Salesforce Bulk API to read Salesforce source data. By default, the PowerCenter Integration Service uses the standard Salesforce API.
Enable PK Chunking	Select to enable the primary key chunking. When you enable primary key chunking, the Bulk API splits the data set into multiple chunks based on the record ID and creates extract queries for each chunk.
PK Chunking Size	The number of records in a chunk. Default is 100000. The maximum value is 250000. Applicable only if you select Enable PK Chunking.
PK Chunking startRow ID	The row ID from where the chunking starts. By default, Salesforce applies chunking from the first record.
PK Chunking Parent Object	Specify the parent object to enable PK chunking for queries on a shared object. For example, for CaseHistory, specify Case as the parent object. PK chunking is supported for shared objects only if the parent object is supported. **Note:** PK Chunking Parent Object name is case sensitive. The name must start with an uppercase letter followed by lowercase letters.

Filtering Source Data

At session run time, the PowerCenter Integration Service generates an SOQL query based on the objects and fields included in the Salesforce source definition. When you configure a session that reads data from a Salesforce source, you can enter a filter condition to filter records from the source.

Enter a filter condition based on the SOQL syntax in the Salesforce documentation. When you enter a filter condition, the PowerCenter Integration Service modifies and overrides the WHERE clause in the SOQL query.

Consider the following guidelines when you enter the filter condition:

- To filter records from a Salesforce source, enter a filter condition for the SOQL Filter Condition session property. For example, enter the following filter condition to read records from the Salesforce Account object that were created before October 30, 2007:

```
CreatedDate < '2007-10-30'
```

- The Order By and Group By clause must be preceded by a filter condition as shown in the following example:

```
id!= null Order By Name
```

The Salesforce API performs SOQL syntax validation at run time. If you enter an invalid filter condition, the session fails.

The session also fails if you enable CDC and one of the following conditions are true:

- You enter an Order By or a Group By clause in the filter condition.
- You enter a LIMIT clause in the filter condition.
- You enter an AND or OR operator in the filter condition, but you do not enclose the query in parentheses. Enclose the query in parentheses as shown in the following example:

```
(Name="Jason" OR Name="Thompson")
```

- You enter a special operator in the filter condition, but you do not enclose the query in parentheses. Enclose the query in parentheses as shown in the following example for the NOT operator:

```
(NOT(Name LIKE 'Jason'))
```

Capturing Deleted and Archived Salesforce Records

The PowerCenter Integration Service can capture active, deleted, and archived records from a Salesforce source object. By default, sessions do not capture deleted and archived records.

When you capture deleted and archived records, you capture deleted data from the recycle bin and archived data from the Salesforce archive. To capture deleted and archived records, configure the Use queryAll session property.

Note: Session fails to fetch the deleted records if the specified time period is more than 30 days.

Configuring a session for changed data capture overrides the Use queryAll property.

Capturing Changed Data

The PowerCenter Integration Service can capture changed data from a Salesforce object that is replicateable and contains CreatedDate and LastModifiedDate fields. If you configure a session to capture changed data from a Salesforce object that is not replicateable or does not contain CreatedDate and LastModifiedDate fields, the session fails. For more information about replicateable objects, see the Salesforce documentation.

When the PowerCenter Integration Service captures deleted data, it returns only the ID of the deleted record and sets the row notification to delete.

Use one of the following methods to capture changed data:

- **Capture changed data continuously.** Configure a session to capture changed data to process data in real-time.
- **Capture changed data for a specific time period.** Configure a session to capture changed data during a particular time period when the data changes.

By default, change data capture is disabled. To enable a particular method, specify the required attributes in the session properties. Configure the attributes for one CDC method. You should not define two different CDC methods in one session.

You can configure the LastModifiedDate or SystemModstamp field as the timestamp that determines when a Salesforce record was last modified.

Configuring a session to perform change data capture overrides the Use queryAll session property.

Related Topics:

- "Using the SystemModstamp or LastModifiedDate Timestamp for Change Data Capture" on page 43

Continuous CDC Sessions

When the PowerCenter Integration Service runs a continuous CDC session, it reads all records in the source object and passes them to the next transformation as rows flagged for insert. After the PowerCenter Integration Service reads all source data, the CDC time limit and flush interval begin.

After the flush interval ends, the PowerCenter Integration Service completes the following tasks to capture changed data for a continuous CDC session:

1. Reads all records created since the initial read and passes them to the next transformation as rows flagged for insert.

2. Reads all records updated since the initial read and passes them to the next transformation as rows flagged for update.
3. Reads all records deleted since the initial read and passes them to the next transformation as rows flagged for delete.

After the PowerCenter Integration Service finishes reading all changed data, the flush interval starts again. The PowerCenter Integration Service stops reading from Salesforce when the CDC time limit ends.

When you configure the session to capture changed data and use source-based commits, the PowerCenter Integration Service commits data to the target based on the source-based commit interval and the flush interval.

For example, you set the CDC time limit to 4,000 seconds, the flush interval to 300 seconds, and the source-based commit interval to 1,000 rows. After the PowerCenter Integration Service reads all the source data, the flush interval begins. The PowerCenter Integration Service captures changed data and commits rows to the target after reading 1,000 rows from the source and after each 300 second flush interval. The PowerCenter Integration Service stops reading from Salesforce after 4,000 seconds.

When you configure the session to use target-based commits, the PowerCenter Integration Service runs the session based on source-based commits. Also, it only commits rows to the target based on the flush interval. It does not commit rows to the target based on the commit interval.

CDC Flush Interval Offset

The CDC flush interval offset is the number of seconds that you want to offset the CDC flush interval.

Configure the flush interval offset to capture real-time data that is submitted within the CDC time limit but not committed by Salesforce within the time limit. A delay might occur when the Salesforce needs to process automatic triggers before it commits the data.

When you configure the session to use a flush interval offset, the PowerCenter Integration Service subtracts the flush interval offset from the flush interval.

For example, you set the flush interval to 300 seconds, and you set the flush interval offset to 2 seconds. The first flush interval starts at 9:00:00 and ends at 9:04:58. Without the flush interval offset, it would have ended at 9:05:00. The second flush interval starts at 9:04:59 and ends at 9:09:57. The third flush interval starts at 9:09:58 and ends at 9:14:56.

Configuring a Continuous CDC Session

Complete the following tasks to capture changed data continuously for sessions that read from replicateable Salesforce objects:

- For each CDC session, complete the configuration steps for real-time sessions.
- Set the time limit and flush interval for change data capture. Optionally set the flush interval offset.

Time-Period Based CDC Sessions

When the PowerCenter Integration Service runs a CDC session for a specific time period, it reads all records in the source object and extracts records that meet the CDC time period criteria.

The PowerCenter Integration Service completes the following steps to capture changed data for a time-period based CDC session:

1. Reads all records created between the CDC start time and end time, and passes them to the next transformation as rows flagged for insert.

2. Reads all records updated between the CDC start time and end time, and passes them to the next transformation as rows flagged for update.
3. Reads all records deleted between the CDC start time and end time, and passes them to the next transformation as rows flagged for delete.

Configuring a Time-Period Based CDC Session

To enable change data capture for a specific time period, define the start and end time for the time period in the session properties.

Rules and Guidelines for Processing a Time-Period Based CDC Session

Use the following rules and guidelines when you run a CDC session for a particular time period:

- The PowerCenter Integration Service validates the formats of the start and end times when you run the session. If either timestamp format is wrong, the session fails.
- The values for the start and end times must be in the past.
- The start time must predate the end time.
- You cannot run the session continuously.
- When you configure the session to capture changed data and to use source-based commits, the PowerCenter Integration Service commits data to the target based on the source-based commit interval. If you configure the session to use target-based commits, the PowerCenter Integration Service runs the session based on source-based commits.

Using the SystemModstamp or LastModifiedDate Timestamp for Change Data Capture

When you run a session that extracts data from a Salesforce source, the PowerCenter Integration Service determines new and updated records based on the Use SystemModstamp for CDC session property. You configure this session property to indicate whether to use the SystemModstamp or LastModifiedDate attribute to determine when a Salesforce record was last modified.

Salesforce updates the LastModifiedDate attribute of a Salesforce record when you update the record. Salesforce updates the SystemModstamp attribute of a Salesforce record when you update the record or a record of a related object.

Salesforce indexes the SystemModstamp attribute, not the LastModifiedDate attribute. To improve session performance, use the SystemModstamp attribute.

Bulk API Source Sessions

The PowerCenter Integration Service can use the Saleforce Bulk API to read data from Salesforce sources. Use the Bulk API to read large amounts of data from Salesforce with a minimal number of API calls. You can use the Bulk API to read data from Salesforce sources with Salesforce API version 25.0 or higher.

Use the Bulk API to read data from individual Salesforce sources. The Bulk API cannot read data from related source objects. To read data from related source objects, use the standard API.

With a Bulk API read, each batch of data can contain up to 1 GB of data in CSV format.

To configure a session to use the Bulk API for Salesforce sources, select the Enable Bulk Query session property. When you select this property, the PowerCenter Integration Service ignores the Use queryAll session property.

The PowerCenter Integration Service ignores the Stop on Error session property for Bulk API source sessions.

Configuring a Session with a Salesforce Target

You can configure the session properties for a Salesforce target on the Mapping tab. Define the properties for each target instance in the session.

The following table describes the session properties that you can configure for a Salesforce target:

Property Name	Description
Treat Insert as Upsert	Upserts any records flagged as insert. By default, the PowerCenter Integration Service treats all records as insert.
Treat Update as Upsert	Upserts any records flagged as update. Select this property when you use the Update Strategy transformation in the mapping or the Treat Source Rows As session property to flag records as update.
Max Batch Size	Maximum number of records the PowerCenter Integration Service writes to a Salesforce target in one batch. Default is 200 records. Not used in Bulk API target sessions.
Set Fields to NULL	Replaces values in the target with null values from the source. By default, the PowerCenter Integration Service does not replace values in a record with null values during an update or upsert operation. It retains the existing values.
Use SFDC Error File	Generates error log files. By default, the PowerCenter Integration Service does not generate error log files. To generate an error log file for a Bulk API target session, select the Monitor Bulk Job Until All Batches Processed session property as well.
Use SFDC Success File	Generates success log files. By default, the PowerCenter Integration Service does not generate success log files. To generate a success log file for a Bulk API target session, select the Monitor Bulk Job Until All Batches Processed session property as well.
SFDC Success File Directory	Directory where the PowerCenter Integration Service stores the success log files. By default, the PowerCenter Integration Service stores the success log files in the $PMTargetFileDir directory. The PowerCenter Integration Service stores the error log files in the $PMBadFileDir directory.
Use Idlookup Field for Upserts	Uses the Salesforce idLookup field to identify target records that need to be upserted. If you do not select this property, use the external ID for the upsert operation. If you do not select this property and do not provide the external ID, the session fails.
Use this ExternalId/ IdLookup field for Upserts	The exact name of the external ID or idLookup field to use for upserts. By default, the PowerCenter Integration Service uses the first external ID or idLookup field in the target. Use this property when you want to use a different field for upserts.
Use SFDC Bulk API	Uses the Salesforce Bulk API to load batch files containing large amounts of data to Salesforce targets. By default, the PowerCenter Integration Service uses the standard Salesforce API.

Property Name	Description
Monitor Bulk Job Until All Batches Processed	Monitors a Bulk API target session. When you select this property, the PowerCenter Integration Service logs the status of each batch in the session log. If you do not select this property, the PowerCenter Integration Service does not generate complete session statistics for the session log. Select this property along with the Use SFDC Success File or Use SFDC Error File session properties to generate success or error logs for the session.
Override Parallel Concurrency with Serial	Instructs the Salesforce Bulk API to write batches to targets serially. By default, the Bulk API writes batches in parallel.
Disable Bulk Success and Error File Creation	Disables the creation of success and error log files for a Bulk API target session. Overrides the Use SFDC Error File and Use SFDC Success File session properties.
Enable Field Truncation Attribute	Allows Salesforce to truncate target data that is larger than the target field. When you select this property, Salesforce truncates overflow data and writes the row to the Salesforce target. By default, the PowerCenter Integration Service writes overflow data to the session error file.
Set Prefix for Success and Error Files	Adds a prefix to the names of the success and error log files. When you add a prefix, the success log file uses the following naming convention: <prefix>_<session name><timestamp>_success.csv. When you add a prefix, the error log file uses the following naming convention: <prefix>_<session name><timestamp>_error.csv.
Enable Hard Deletes for Bulk API	Permanently deletes rows from Salesforce targets in a Bulk API target session.
Set the Location of the Bulk Error Files	Directory for Bulk API target error log files.
Set the Interval for Polling Bulk Job Status	Number of seconds the PowerCenter Integration Service waits before polling Salesforce for information about a Bulk API target session. Enter a positive integer. By default, the PowerCenter Integration Service polls every 10 seconds.
Assignment Rule Selection	Applicable only to Salesforce Case and Lead target objects using the standard API. Assignment rule to reassign attributes in records when inserting, updating, or upserting records: - None. Select to use no assignment rule. Default is None. - Default. Select to use the default assignment rule set for the organization. - Custom. Select to specify and use a custom assignment rule. **Note:** You cannot use Assignment Rule Selection for Bulk API.
Specify Assignment Rule	For custom assignment rule. Enter a valid assignment rule.

Configuring the Upsert Target Operation

The Salesforce upsert operation creates a new record or updates an existing record in a Salesforce object. You must provide one of the following types of fields to upsert records to a Salesforce object:

- **External ID field.** You can use a custom Salesforce field to uniquely identify each record in a Salesforce object. You can create a custom external ID field for each object in Salesforce. You can view the properties of a Salesforce target definition in the PowerCenter Designer to see if the object includes an external ID field.
- **idLookup field.** You can use a Salesforce idLookup field to identify each record in a Salesforce object. Salesforce creates idLookup fields for each standard Salesforce object. For example, the Email field is an idLookup field for the Contact object. Custom Salesforce objects do not contain an idLookup field. For more information about idLookup fields, see the Salesforce documentation.

A Salesforce target object may have multiple external ID or Idlookup fields. By default, the PowerCenter Integration Service uses the first external ID or Idlookup field it encounters. However, you can specify the external ID or Idlookup field to use for the upsert operation in the session properties.

To configure the upsert operation for a session that writes to a Salesforce target:

1. Map the external ID or idLookup field from the source to the target in the mapping. If you are using an external ID, map the external ID to the external ID field in the Salesforce target object. If you are using an idLookup field, map the field to the appropriate target field. For example, map the email source field to the Email field in the Salesforce Contact object target.
2. Configure the Treat Insert as Upsert or Treat Update as Upsert session properties to configure a Salesforce session to upsert records.
3. To use the idLookup field instead of an external ID field, enable the Use IdLookup Field for Upserts session property. By default, the PowerCenter Integration Service uses the external ID for upserts. You can configure the session to override the external ID, and use the idLookup instead.
4. To specify which external ID or Idlookup field to use, enter the external ID or Idlookup field name in the Use this ExternalId/Idlookup Field for Upserts session property.

   Note: If you do not enter the name of the external ID or Idlookup field, the PowerCenter Integration Service selects the first external ID or Idlookup field it encounters. If you specify a field that is not an external ID or Idlookup field, or you misspell the field name, the session fails.

Configuring the Maximum Batch Size

The PowerCenter Integration Service writes data to a Salesforce target as a batch. The Max Batch Size attribute in the session properties determines the maximum number of records the PowerCenter Integration Service can write to a Salesforce target in a batch. The Salesforce service can receive a maximum of 200 records in a single insert, update, or delete call.

To minimize the number of calls made to the Salesforce service, each batch should contain the maximum number of records as configured in the Max Batch Size property. To optimize session performance, use the default Max Batch Size of 200 and tune the DTM Buffer Size session property so that buffer blocks contain multiples of 200 records.

Related Topics:

- "Tuning the DTM Buffer Size" on page 52

Handling Null Values in Update and Upsert Operations

By default, the PowerCenter Integration Service does not replace existing values in a Salesforce record with null values from the source during an update or upsert operation. To replace existing values with null values, configure the Set Fields to NULL session property for the Salesforce target.

You cannot set the value of an external ID field in a Salesforce target to null. The session fails if you enable the Set Fields to NULL session property and the session tries to replace an existing value for an external ID field with a null value.

Disassociate a Child Object

You can disassociate a custom child object from a standard parent object by performing an upsert operation with the standard API. When you disassociate a child object, you detach the child object from the parent object.

To disassociate a custom child object from a standard parent object perform the following tasks:

- Use the Salesforce standard API.
- Configure the session to perform upserts with the Treat Insert as Upsert or the Treat Update as Upsert session property.
- Configure the session to perform null updates with the Set Fields to Null session property.
- In the same upsert row, pass a NULL value to the following fields:
  - The external ID field of the parent object.
  - Any field in the child object.

For example, you have a custom Contact object with a child relationship to a standard Account object. To disassociate the custom Contact object from the standard Account object, you can use the standard API to upsert a row. The row includes a null value for the external ID of the Account object and a null value for any field in the Contact object.

Logging PowerExchange for Salesforce Session Details

The PowerCenter Integration Service can generate record-level logs for each session that writes to a Salesforce target. The PowerCenter Integration Service can generate the following types of logs:

- **Success log.** The success log contains an entry for each record that successfully loads into the Salesforce target. Each entry contains the values loaded for all fields of the record, including the Salesforce ID. Use this file understand what data is loaded into the Salesforce target. You can use the success log as an audit trail for all operations performed.
- **Error log.** The error log contains an entry for each data error. Each log entry contains the values for all fields of the record and the error message. The error log displays error messages from Salesforce and PowerCenter. Use this file to understand why records did not load into the Salesforce target.

The PowerCenter Integration Service inserts quotes around commas and quotes included in the data.

Note: The PowerExchange for Salesforce success and error logs are different from the PowerCenter session logs. The PowerExchange for Salesforce success and error logs contain record-level details that are specific to sessions with Salesforce targets. PowerCenter session logs contain information about the tasks performed by the PowerCenter Integration Service, general session errors, and load summary and transformation statistics for the session.

The PowerCenter Integration Service uses the following naming conventions for the success and error log files:

Type of Log File	Naming Convention
Success log file	<session name><timestamp>_success.csv
Error log file	<session name><timestamp>_error.csv

To configure the PowerCenter Integration Service to generate success and error logs for a session that writes to Salesforce target, configure the Use SFDC Error File, Use SFDC Success File, and SFDC Success File Directory session properties.

You can add a prefix for the success or error log files for a session. When you configure a prefix, the PowerCenter Integration Service adds the prefix to the log file names as follows:

Type of Log File	Naming Convention with Prefix
Success log file	<prefix>_<session name><timestamp>_success.csv
Error log file	<prefix>_<session name><timestamp>_error.csv

To add a prefix to the success or error log, use the Set Prefix for Success and Error Files session property.

Sample Success Log

The following table shows an excerpt of a success log:

ID	NAME	ACCOUNT_NO	EXT_ACCOUNT_NO__C	STATUS
0017000000NYre4AAD	JumpCo	76543	666	Item Created

The Status field indicates whether PowerCenter Integration Service created, updated, or deleted the record.

Sample Error Log

The following table shows an excerpt of an error log:

NAME	ACCOUNT_NO	ANNUAL_REV	EXT_ACCOUNT_NO_C	ERROR
Company1	76544	8 million	555	Error converting value to correct data type: For input string: "8 million"
Company2	76545	-	444	Duplicate external id specified: 444
Company3	76546	3.70E+10	444	Duplicate external id specified: 444

Override an External ID with an idLookup for Upserts

The PowerCenter Integration Service can use the external ID or idLookup fields when performing an upsert operation on a Salesforce target. The PowerCenter Integration Service uses one of these IDs to identify records in the Salesforce target to upsert. By default, the PowerCenter Integration Service uses the external ID for upserts. You can configure the session to override the external ID, and use the idLookup instead.

Bulk API Target Sessions

The PowerCenter Integration Service can use the Salesforce Bulk API to write data to Salesforce targets. Use the Bulk API to write large amounts of data to Salesforce with a minimal number of API calls. You can use the Bulk API to write data to Salesforce targets with Salesforce API version 20.0 or higher. With a Bulk API write, each batch of data can contain up to 10,000 records or one million characters of data in CSV format.

You can configure a Bulk API target session to load batches at the same time or serially. You can also monitor the progress of batches in the session log, and create success and error logs for row-level information.

To configure a session to use the Bulk API for Salesforce targets, select the Use SFDC Bulk API session property. When you select this property, the PowerCenter Integration Service ignores the Max Batch Size session property.

The PowerCenter Integration Service ignores the Stop on Error session property for Bulk API target sessions.

When you configure error logging for a relational database, the PowerCenter Integration Service does not load the error messages to PMERR tables for the Salesforce target.

Before you run a Bulk API target session, configure the Java temporary directory on the PowerCenter Integration Service machine.

Monitor a Bulk API Target Session

The PowerCenter Integration Service can monitor a Salesforce Bulk API target session. Configure a Bulk API target session for monitoring when you want to review detailed session statistics or when you want to generate a success or error log for the session.

When the PowerCenter Integration Service monitors a Bulk API target session, it requests the status of each batch from the Salesforce service. The PowerCenter Integration Service repeats the request every 10 seconds until all batches complete, and writes the response from the Salesforce service in the session log. The PowerCenter Integration Service also generates a success log or error log if requested.

By default, the PowerCenter Integration Service does not monitor Bulk API target sessions. Without monitoring, the session log contains information about batch creation, but does not contain details about batch processing or complete session statistics. In addition, the PowerCenter Integration Service does not generate success or error logs for the session.

Note: The PowerCenter Integration Service performs additional API calls when monitoring a Bulk API target session. To reduce the number of API calls the PowerCenter Integration Service makes, do not monitor the session. For more information about batch processing, use the batch IDs in the session log to access Salesforce statistics.

To configure a session for monitoring, select the Monitor Bulk Job Until All Batches Processed session property.

To see how many records are successfully loaded for each batch, set the Override Tracing session property to a value other than “None.”

Bulk API Target Load Type

The Salesforce service can perform a parallel load or serial load for Bulk API targets. By default, it performs a parallel load.

In a parallel load, the Salesforce service writes batches to session targets at the same time. It processes each batch as soon as possible. In a serial load, the Salesforce service writes batches to targets in the order it receives them. It processes the entire contents of each batch before proceeding to the next batch.

Use a parallel load to increase performance when you are not concerned about the target load order. Use a serial load when you want to preserve the target load order, such as during an upsert load.

To configure the Bulk API target session for a serial load, select the Override Parallel Concurrency with Serial session property.

Bulk API Target Success Files and Error Files

When the PowerCenter Integration Service monitors a Salesforce Bulk API target session, it can generate Bulk API success logs and error logs. Success logs and error logs are CSV files that contain row-level details from the Salesforce service.

When the Salesforce service completes all batch processing for a Bulk API target session, the PowerCenter Integration Service requests success information and error information. The PowerCenter Integration Service uses this information to generate success logs and error logs.

The following table describes the locations and naming conventions for Bulk API target success log files and Bulk API target error log files:

Log Type	Location	Naming Convention
Bulk API success log	$PMTargetFileDir	<session name>_<timestamp>_bulk_success.csv
Bulk API error log	$PMBadFileDir	<session name>_<timestamp>_bulk_error.csv

To generate a success log, select the Use SFDC Success File session property. To generate an error log, select the Use SFDC Error File session property. To ensure the PowerCenter Integration Service generates logs, select the Monitor Bulk Job Until All Batches Processed session property to enable monitoring.

You can add a prefix for the Bulk API success log file and error log files for a session. When you configure a prefix, the PowerCenter Integration Service adds the prefix to the log file names as follows:

Log Type	Location	Naming Convention with Prefix
Bulk API success log	$PMTargetFileDir	<prefix>_<session name>_<timestamp>_bulk_success.csv
Bulk API error log	$PMBadFileDir	<prefix>_<session name>_<timestamp>_bulk_error.csv

To add a prefix to the success log file or error log file, use the Set Prefix for Success and Error Files session property.

Bulk API Success Log

The Bulk API success log contains the following columns:

Column Name	Description
JobId	Job ID.
BatchId	Target batch ID.
ID	Salesforce ID.

Column Name	Description
Success	Indicates the row was successfully written to the target. For all rows in the Bulk API success log, default is True.
Created	Indicates whether the row inserted or updated in the target: - True. The row was inserted. - False. The row was updated.
Error	Error message associated with the row. No errors should appear in the Bulk API success log.

In addition to these columns, the Bulk API success log includes the entire source row. You can use the error message to correct the error in the source row.

The following table shows an excerpt of a Bulk API success log:

JobId	BatchId	ID	Success	Created	Error	ID	Name
750i000000	751i000000	001T000000	TRUE	FALSE	-	2340A1	CAM Water
13Q7SAAU	1Jzz7AAC	HsvMdIAJ					
750i000000	751i000000	001T000000	TRUE	TRUE	-	2340A2	RDD Electric
13Q7SAAU	1Jzz7AAC	HsvMeIAJ					
750i000000	751i000000	001T000000	TRUE	FALSE	-	2340A3	NXY Gas
13Q7SAAU	1Jzz7AAC	HsvMfIAJ					

For more information about the Bulk API success log, see the Salesforce documentation.

Bulk API Error Log

The Bulk API error log contains the following columns:

Column Name	Description
JobId	Job ID.
BatchId	Target batch ID.
ID	Salesforce ID. This field might not be populated.
Success	Indicates the row was successfully written to the target. For all rows in the Bulk API error log, default is False.
Created	Not relevant for error rows.
Error	Error message associated with the row.

In addition to these columns, the Bulk API error log includes the entire source row. You can use the error message to correct the error in the source row.

The following table shows an excerpt of a Bulk API error log:

JobId	BatchId	ID	Success	Created	Error	ID	Name
750i000000 13Q7SAAU	751i000000 1Jzz7AAC	-	FALSE	FALSE	MALFORMED_ID:Account ID: id value of incorrect type: ÃƒÂ¯Ã'Â»Ã'Â¿	Ã¯Â»Â¿	MDM Water
750i000000 13Q7SAAU	751i000000 1Jzz7AAC	-	FALSE	FALSE	MALFORMED_ID:Account ID: id value of incorrect type: QHP1NÃ£	QHP1NÃ£	RAA Electric

For more information about the Bulk API error log, see the Salesforce documentation.

Hard Deletes with Bulk API Targets

When you use the Salesforce Bulk API, you can configure the session to permanently delete rows from Salesforce targets.

By default, when you use the Bulk API to delete data from Salesforce targets, the PowerCenter Integration Service copies the deleted rows to the recycle bin. You can retrieve the deleted rows until you empty the recycle bin. Until then, the deleted rows occupy space on the hard disk.

To permanently delete rows, you can configure a Bulk API target session to perform a hard delete. With a hard delete, the PowerCenter Integration Service permanently deletes rows, bypassing the recycle bin. You cannot recover data that the PowerCenter Integration Service deletes with the hard delete option.

Configuring a Session for Optimal Performance

You can increase the performance of PowerExchange for Salesforce by editing the mapping or session. Configure the following options to increase the session performance:

- **DTM buffer size**. You can increase or decrease the value of DTM Buffer Size to specify the amount of memory the PowerCenter Integration Service uses as DTM buffer memory.
- **Precision of string fields.** Modify the precision of fields that map to the string transformation datatype.

For more information about these parameters see the *PowerCenter Performance Tuning Guide*.

Tuning the DTM Buffer Size

The DTM Buffer Size determines the size of the buffer blocks that the PowerCenter Integration Service uses to move data from sources to targets. When a target definition receives a buffer block, the PowerCenter Integration Service creates one or more batches to send the records to the Salesforce service.

For example, if Max Batch Size is set to 200 and a buffer block contains 200 records, the PowerCenter Integration Service sends one batch to the Salesforce target containing 200 records. If a buffer block contains 500 records, the PowerCenter Integration Service sends three batches containing 200, 200, and 100 records each.

The performance of a session that writes to a Salesforce target may be slower than a similar session that writes to a relational database because the Salesforce service runs on top of a relational database.

To optimize the performance of a session for a Salesforce target, tune the DTM Buffer Size session property so that buffer blocks contain multiples of 200 records.

Modifying the Precision of String Fields

To improve performance, you can reduce the precision to the anticipated size of the source or target data. For example, the precision of a field of the String datatype in a Salesforce source definition is 64 KB. However, this field contains data with a maximum size of 1 KB. You can reduce the precision for the field in the source definition to 1 KB so that the PowerCenter Integration Service processes less data for the field.

Edit the Salesforce source or target definition to change the precision of fields that map to the String transformation datatype in PowerCenter. Do not change the precision of fields that do not map to the String transformation datatype in PowerCenter.

APPENDIX A

Datatype Reference

This appendix includes the following topics:

Datatype Reference Overview

When the PowerCenter Integration Service reads data from a Salesforce object, it converts each Salesforce datatype to a compatible PowerCenter transformation datatype. When the PowerCenter Integration Service loads data into a Salesforce object, it converts each PowerCenter transformation datatype to a compatible Salesforce datatype.

PowerExchange for Salesforce uses the following datatypes:

- **Salesforce datatypes.** Salesforce datatypes appear in Salesforce definitions in a mapping.
- **Transformation datatypes.** Transformation datatypes are generic datatypes that PowerCenter uses during the transformation process. The transformation datatypes appear in PowerCenter transformations.

Salesforce and Transformation Datatypes

When the PowerCenter Integration Service reads data from a Salesforce source, it converts the data in the data fields to transformation datatypes used in the Application Source Qualifier. When writing data to a Salesforce target, the PowerCenter Integration Service converts the data based on the native datatypes in the target objects.

The following table shows the conversion between Salesforce datatypes and transformation datatypes:

Salesforce Datatype	Range and Description	Transformation Datatype	Range and Description
AnyType	Polymorphic data type that returns string, picklist, reference, boolean, currency, integer, double, percent, ID, date, datetime, URL, or email data.	String	1 to 104,857,600 characters
Base64	Base64-encoded arbitrary binary data.	String	1 to 104,857,600 characters
Boolean	Boolean (true/false) values.	Integer	Precision 10, scale 0
Currency	Currency values.	Decimal	Precision 1 to 28 digits, scale 0 to 28
DataCategoryGroupReference	Types of category groups and category unique names.	String	1 to 104,857,600 characters
Date	Date values.	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)
DateTime	Date and time values.	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)
Double	Double values.	Decimal	Precision 1 to 28 digits, scale 0 to 28
Email	Email addresses.	String	1 to 104,857,600 characters
Encrypted String	Encrypted text fields contain any combination of letters, numbers, or symbols that are stored in encrypted form.	String	1 to 104,857,600 characters
Geolocation	A compound field to specify a location by latitude and longitude.	Double	Precision 1 to 28 digits, scale 0 to 28. PowerCenter Integration Service converts Geolocation into two Double values, one each for latitude and longitude.
ID	Primary key for a Salesforce object.	String	1 to 104,857,600 characters
Int	Integer values.	Integer	Precision 10, scale 0
Multipicklist	Multiple-selection picklists, which provide a set of enumerated values that you can select multiple values from.	String	1 to 104,857,600 characters
Percent	Percentage values.	Decimal	Precision 1 to 28 digits, scale 0 to 28
Picklist	Single-selection picklists, which provide a set of enumerated values that you can select one value from.	String	1 to 104,857,600 characters

Salesforce Datatype	Range and Description	Transformation Datatype	Range and Description
Reference	Cross-reference to another Salesforce object.	String	1 to 104,857,600 characters
String	String values.	String	1 to 104,857,600 characters
Textarea	String that appears as a multiple-line text field.	String	1 to 104,857,600 characters
Time	Time values.	Date/Time	Precision 1 to 28 digits, scale 0 to 28. PowerCenter Integration Service adds the current date when it converts the Salesforce time datatype to the Date/Time transformation datatype. The PowerCenter Integration Service trims the date when it converts the Date/Time transformation datatype to the Salesforce time datatype.
Url	URL values.	String	1 to 104,857,600 characters

Note: After you import a Salesforce definition, you can change the precision of fields that map to the String transformation datatype in PowerCenter. In some cases, you might want to decrease the precision to improve performance. Do not change the precision of fields that do not map to the String transformation datatype in PowerCenter.

APPENDIX B

Glossary

Change Data Capture (CDC)

An option that enables PowerExchange for Salesforce to process changed data. You can capture changed data for replicateable objects that have the CreatedDate and LastModifiedDate fields.

external ID

A record ID generated by a system outside of Salesforce. The PowerCenter Integration Service uses the external ID field to identify records in a Salesforce object if the object is configured with an external ID field. You indicate whether a custom field is an external ID field in Salesforce.

primary Salesforce object

The primary object in Salesforce that you import to create a source or target definition, or transformation in PowerCenter.

related Salesforce object

A Salesforce object that is related to a primary Salesforce object. When you import a Salesforce definition based on a primary Salesforce object, you can also import fields of related objects. The Designer determines related objects based on the relationships defined in Salesforce.

replicateable object

A Salesforce object that can be duplicated in another environment. Every Salesforce object has a replicateable flag that determines if an object can be duplicated in another environment. PowerExchange for Salesforce uses the flag to determine if it can replicate the Salesforce object.

Salesforce objects

Subject areas in Salesforce, such as Account, Asset, Case Solution, and Lead. Salesforce objects usually correspond to tabs in the Salesforce user interface. For example, you can add accounts in the Salesforce Accounts tab. You import standard and custom Salesforce objects into PowerCenter as transformations, and source and target definitions.

Salesforce Sandbox

A special testing or development Salesforce environment.

Sforce Object Query Language (SOQL)

Proprietary Salesforce language used to construct a query in a Salesforce query call. At run time, the PowerCenter Integration Service generates an SOQL query based on the objects and fields included in the mapping.

upsert

A Salesforce operation that uses a custom field designated as the external ID field to determine whether it should create a new record or update an existing one when loading data into a Salesforce target. You can configure the session to upsert records only if the Salesforce target object includes a custom field designated as the external ID field.

INDEX

F

H

I

J

L

M

N

O

P

R

S

T

U

V