



Informatica PowerExchange for Salesforce
(Version 9.5.1.4.1)

User Guide for PowerCenter

Copyright (c) 2006-2014 Informatica Corporation. All rights reserved.

This software and documentation contain proprietary information of Informatica Corporation and are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright law. Reverse engineering of the software is prohibited. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica Corporation. This Software may be protected by U.S. and/or international Patents and other Patents Pending.

Use, duplication, or disclosure of the Software by the U.S. Government is subject to the restrictions set forth in the applicable software license agreement and as provided in DFARS 227.7202-1(a) and 227.7702-3(a) (1995), DFARS 252.227-7013^(c)(1)(ii) (OCT 1988), FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as applicable.

The information in this product or documentation is subject to change without notice. If you find any problems in this product or documentation, please report them to us in writing.

Informatica, Informatica Platform, Informatica Data Services, PowerCenter, PowerCenterRT, PowerCenter Connect, PowerCenter Data Analyzer, PowerExchange, PowerMart, Metadata Manager, Informatica Data Quality, Informatica Data Explorer, Informatica B2B Data Transformation, Informatica B2B Data Exchange Informatica On Demand, Informatica Identity Resolution, Informatica Application Information Lifecycle Management, Informatica Complex Event Processing, Ultra Messaging and Informatica Master Data Management are trademarks or registered trademarks of Informatica Corporation in the United States and in jurisdictions throughout the world. All other company and product names may be trade names or trademarks of their respective owners.

Portions of this software and/or documentation are subject to copyright held by third parties, including without limitation: Copyright DataDirect Technologies. All rights reserved. Copyright © Sun Microsystems. All rights reserved. Copyright © RSA Security Inc. All Rights Reserved. Copyright © Ordinal Technology Corp. All rights reserved. Copyright © Aandacht c.v. All rights reserved. Copyright Genivia, Inc. All rights reserved. Copyright Isomorphic Software. All rights reserved. Copyright © Meta Integration Technology, Inc. All rights reserved. Copyright © Intalio. All rights reserved. Copyright © Oracle. All rights reserved. Copyright © Adobe Systems Incorporated. All rights reserved. Copyright © DataArt, Inc. All rights reserved. Copyright © ComponentSource. All rights reserved. Copyright © Microsoft Corporation. All rights reserved. Copyright © Rogue Wave Software, Inc. All rights reserved. Copyright © Teradata Corporation. All rights reserved. Copyright © Yahoo! Inc. All rights reserved. Copyright © Glyph & Cog, LLC. All rights reserved. Copyright © Thinkmap, Inc. All rights reserved. Copyright © Clearpace Software Limited. All rights reserved. Copyright © Information Builders, Inc. All rights reserved. Copyright © OSS Nokalva, Inc. All rights reserved. Copyright Edifecs, Inc. All rights reserved. Copyright Cleo Communications, Inc. All rights reserved. Copyright © International Organization for Standardization 1986. All rights reserved. Copyright © ej-technologies GmbH. All rights reserved. Copyright © Jaspersoft Corporation. All rights reserved. Copyright © is International Business Machines Corporation. All rights reserved. Copyright © yWorks GmbH. All rights reserved. Copyright © Lucent Technologies. All rights reserved. Copyright (c) University of Toronto. All rights reserved. Copyright © Daniel Veillard. All rights reserved. Copyright © Unicode, Inc. Copyright IBM Corp. All rights reserved. Copyright © MicroQuill Software Publishing, Inc. All rights reserved. Copyright © PassMark Software Pty Ltd. All rights reserved. Copyright © LogiXML, Inc. All rights reserved. Copyright © 2003-2010 Lorenzi Davide, All rights reserved. Copyright © Red Hat, Inc. All rights reserved. Copyright © The Board of Trustees of the Leland Stanford Junior University. All rights reserved. Copyright © EMC Corporation. All rights reserved. Copyright © Flexera Software. All rights reserved. Copyright © Jinfonet Software. All rights reserved. Copyright © Apple Inc. All rights reserved. Copyright © Telerik Inc. All rights reserved. Copyright © BEA Systems. All rights reserved. Copyright © PDFlib GmbH. All rights reserved. Copyright © Orientation in Objects GmbH. All rights reserved. Copyright © Tanuki Software, Ltd. All rights reserved. Copyright © Ricebridge. All rights reserved. Copyright © Sencha, Inc. All rights reserved.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>), and/or other software which is licensed under various versions of the Apache License (the "License"). You may obtain a copy of these Licenses at <http://www.apache.org/licenses/>. Unless required by applicable law or agreed to in writing, software distributed under these Licenses is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the Licenses for the specific language governing permissions and limitations under the Licenses.

This product includes software which was developed by Mozilla (<http://www.mozilla.org/>), software copyright The JBoss Group, LLC, all rights reserved; software copyright © 1999-2006 by Bruno Lowagie and Paulo Soares and other software which is licensed under various versions of the GNU Lesser General Public License Agreement, which may be found at <http://www.gnu.org/licenses/lgpl.html>. The materials are provided free of charge by Informatica, "as-is", without warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

The product includes ACE(TM) and TAO(TM) software copyrighted by Douglas C. Schmidt and his research group at Washington University, University of California, Irvine, and Vanderbilt University, Copyright (©) 1993-2006, all rights reserved.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (copyright The OpenSSL Project. All Rights Reserved) and redistribution of this software is subject to terms available at <http://www.openssl.org> and <http://www.openssl.org/source/license.html>.

This product includes Curl software which is Copyright 1996-2013, Daniel Stenberg, <daniel@haxx.se>. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://curl.haxx.se/docs/copyright.html>. Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

The product includes software copyright 2001-2005 (©) MetaStuff, Ltd. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://www.dom4j.org/license.html>.

The product includes software copyright © 2004-2007, The Dojo Foundation. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://dojotoolkit.org/license>.

This product includes ICU software which is copyright International Business Machines Corporation and others. All rights reserved. Permissions and limitations regarding this software are subject to terms available at <http://source.icu-project.org/repos/icu/icu/trunk/license.html>.

This product includes software copyright © 1996-2006 Per Bothner. All rights reserved. Your right to use such materials is set forth in the license which may be found at <http://www.gnu.org/software/kawa/Software-License.html>.

This product includes OSSP UUID software which is Copyright © 2002 Ralf S. Engelschall, Copyright © 2002 The OSSP Project Copyright © 2002 Cable & Wireless Deutschland. Permissions and limitations regarding this software are subject to terms available at <http://www.opensource.org/licenses/mit-license.php>.

This product includes software developed by Boost (<http://www.boost.org/>) or under the Boost software license. Permissions and limitations regarding this software are subject to terms available at http://www.boost.org/LICENSE_1_0.txt.

This product includes software copyright © 1997-2007 University of Cambridge. Permissions and limitations regarding this software are subject to terms available at <http://www.pcre.org/license.txt>.

This product includes software copyright © 2007 The Eclipse Foundation. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://www.eclipse.org/org/documents/epl-v10.php> and at <http://www.eclipse.org/org/documents/edl-v10.php>.

This product includes software licensed under the terms at <http://www.tcl.tk/software/tcltk/license.html>, <http://www.bosrup.com/web/overlib/?License>, <http://www.stlport.org/doc/license.html>, <http://asm.ow2.org/license.html>, <http://www.cryptix.org/LICENSE.TXT>, <http://hsqldb.org/web/hsqLicense.html>, <http://httpunit.sourceforge.net/doc/license.html>, <http://jung.sourceforge.net/license.txt>, http://www.gzip.org/zlib/zlib_license.html, <http://www.openldap.org/software/release/>

license.html, <http://www.libssh2.org>, <http://slf4j.org/license.html>, <http://www.sente.ch/software/OpenSourceLicense.html>, <http://fusesource.com/downloads/license-agreements/fuse-message-broker-v-5-3- license-agreement>; <http://antlr.org/license.html>; <http://aopalliance.sourceforge.net/>; <http://www.bouncycastle.org/licence.html>; <http://www.jgraph.com/jgraphdownload.html>; <http://www.jcraft.com/jsch/LICENSE.txt>; http://jotm.objectweb.org/bsd_license.html; . <http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231>; <http://www.slf4j.org/license.html>; <http://nanoxml.sourceforge.net/orig/copyright.html>; <http://www.json.org/license.html>; <http://forge.ow2.org/projects/javaservice/>, <http://www.postgresql.org/about/licence.html>, <http://www.sqlite.org/copyright.html>, <http://www.tcl.tk/software/tcltk/license.html>, <http://www.jaxen.org/faq.html>, <http://www.jdom.org/docs/faq.html>, <http://www.slf4j.org/license.html>; <http://www.iodbc.org/dataspace/iodbc/wiki/ODBC/License>; <http://www.keplerproject.org/md5/license.html>; <http://www.toedter.com/en/jcalendar/license.html>; <http://www.edankert.com/bounce/index.html>; <http://www.net-snmp.org/about/license.html>; <http://www.openmdx.org/#FAQ>; http://www.php.net/license/3_01.txt; <http://srp.stanford.edu/license.txt>; <http://www.schneier.com/blowfish.html>; <http://www.jmock.org/license.html>; <http://xsom.java.net>; <http://benalman.com/about/license/>; <https://github.com/CreateJS/EaselJS/blob/master/src/easeljs/display/Bitmap.js>; <http://www.h2database.com/html/license.html#summary>; <http://jsoncpp.sourceforge.net/LICENSE>; <http://jdbc.postgresql.org/license.html>; <http://protobuf.googlecode.com/svn/trunk/src/google/protobuf/descriptor.proto>; <https://github.com/rantav/hector/blob/master/LICENSE>; <http://web.mit.edu/Kerberos/krb5-current/doc/mitK5license.html>; <http://jibx.sourceforge.net/jibx-license.html>; and <https://github.com/lyokato/libgeohash/blob/master/LICENSE>.

This product includes software licensed under the Academic Free License (<http://www.opensource.org/licenses/afl-3.0.php>), the Common Development and Distribution License (<http://www.opensource.org/licenses/cddl1.php>) the Common Public License (<http://www.opensource.org/licenses/cpl1.0.php>), the Sun Binary Code License Agreement Supplemental License Terms, the BSD License (<http://www.opensource.org/licenses/bsd-license.php>), the new BSD License (<http://opensource.org/licenses/BSD-3-Clause>), the MIT License (<http://www.opensource.org/licenses/mit-license.php>), the Artistic License (<http://www.opensource.org/licenses/artistic-license-1.0>) and the Initial Developer's Public License Version 1.0 (<http://www.firebirdsql.org/en/initial-developer-s-public-license-version-1-0/>).

This product includes software copyright © 2003-2006 Joe Walnes, 2006-2007 XStream Committers. All rights reserved. Permissions and limitations regarding this software are subject to terms available at <http://xstream.codehaus.org/license.html>. This product includes software developed by the Indiana University Extreme! Lab. For further information please visit <http://www.extreme.indiana.edu/>.

This product includes software Copyright (c) 2013 Frank Balluffi and Markus Moeller. All rights reserved. Permissions and limitations regarding this software are subject to terms of the MIT license.

This Software is protected by U.S. Patent Numbers 5,794,246; 6,014,670; 6,016,501; 6,029,178; 6,032,158; 6,035,307; 6,044,374; 6,092,086; 6,208,990; 6,339,775; 6,640,226; 6,789,096; 6,823,373; 6,850,947; 6,895,471; 7,117,215; 7,162,643; 7,243,110; 7,254,590; 7,281,001; 7,421,458; 7,496,588; 7,523,121; 7,584,422; 7,676,516; 7,720,842; 7,721,270; 7,774,791; 8,065,266; 8,150,803; 8,166,048; 8,166,071; 8,200,622; 8,224,873; 8,271,477; 8,327,419; 8,386,435; 8,392,460; 8,453,159; 8,458,230; and RE44,478, International Patents and other Patents Pending.

DISCLAIMER: Informatica Corporation provides this documentation "as is" without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of noninfringement, merchantability, or use for a particular purpose. Informatica Corporation does not warrant that this software or documentation is error free. The information provided in this software or documentation may include technical inaccuracies or typographical errors. The information in this software and documentation is subject to change at any time without notice.

NOTICES

This Informatica product (the "Software") includes certain drivers (the "DataDirect Drivers") from DataDirect Technologies, an operating company of Progress Software Corporation ("DataDirect") which are subject to the following terms and conditions:

1. THE DATADIRECT DRIVERS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.
2. IN NO EVENT WILL DATADIRECT OR ITS THIRD PARTY SUPPLIERS BE LIABLE TO THE END-USER CUSTOMER FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR OTHER DAMAGES ARISING OUT OF THE USE OF THE ODBC DRIVERS, WHETHER OR NOT INFORMED OF THE POSSIBILITIES OF DAMAGES IN ADVANCE. THESE LIMITATIONS APPLY TO ALL CAUSES OF ACTION, INCLUDING, WITHOUT LIMITATION, BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, STRICT LIABILITY, MISREPRESENTATION AND OTHER TORTS.

Part Number: PWX-SFU-95141-000-0001

Table of Contents

| | |
|---|-----------|
| Preface | iv |
| Informatica Resources. | iv |
| Informatica My Support Portal. | iv |
| Informatica Documentation. | iv |
| Informatica Web Site. | iv |
| Informatica How-To Library. | v |
| Informatica Knowledge Base. | v |
| Informatica Support YouTube Channel. | v |
| Informatica Marketplace. | v |
| Informatica Velocity. | v |
| Informatica Global Customer Support. | v |
| Chapter 1: Understanding PowerExchange for Salesforce..... | 1 |
| Understanding PowerExchange for Salesforce Overview. | 1 |
| PowerCenter and Salesforce Integration. | 2 |
| Designer and Salesforce Integration. | 2 |
| PowerCenter Integration Service and Salesforce Integration. | 2 |
| Chapter 2: Installation and Configuration..... | 4 |
| Installation and Configuration Overview. | 4 |
| Prerequisites. | 4 |
| Installing and Configuring PowerExchange for Salesforce. | 4 |
| Upgrading PowerExchange for Salesforce. | 4 |
| Plug-in Registration. | 5 |
| HTTP Proxy Options. | 5 |
| Configuring HTTP Proxy Options for the PowerCenter Client. | 5 |
| Configuring HTTP Proxy Options for the PowerCenter Integration Service. | 6 |
| Java Requirements for Bulk API Target Sessions. | 7 |
| Uninstalling PowerExchange for Salesforce. | 7 |
| Release Information. | 7 |
| New Features. | 7 |
| Changed Behavior. | 9 |
| Chapter 3: Salesforce Sources and Targets..... | 14 |
| Salesforce Sources and Targets Overview. | 14 |
| Importing Fields from Related Salesforce Objects. | 14 |
| Rules and Guidelines for Importing Fields from Related Salesforce Objects. | 15 |
| Salesforce Sources. | 16 |
| Time Zones for Salesforce Sources. | 16 |
| Time Conversion from Salesforce Sources. | 16 |

| | |
|--|-----------|
| Salesforce Targets. | 16 |
| Time Zones for Salesforce Targets. | 17 |
| Determining Possible Update Strategies for a Salesforce Target. | 17 |
| Rules and Guidelines for the Salesforce Target Update Strategy. | 17 |
| Importing a Salesforce Source or Target Definition. | 18 |
| Chapter 4: Salesforce Lookup Transformation. | 20 |
| Salesforce Lookup Transformation Overview. | 20 |
| Salesforce Lookup Components. | 21 |
| Salesforce Lookup Ports. | 22 |
| Lookup Ports. | 22 |
| Pass-Through Ports. | 23 |
| LKP_FILTER Port. | 23 |
| LKP_MATCHIDX Port. | 23 |
| Salesforce Lookup Query. | 24 |
| Creating a Salesforce Lookup Transformation. | 24 |
| Chapter 5: Salesforce Merge Transformation. | 26 |
| Salesforce Merge Transformation Overview. | 26 |
| Sample Salesforce Merge Transformation. | 27 |
| Salesforce Merge Components. | 27 |
| Salesforce Merge Ports. | 28 |
| Salesforce Object Attribute Ports. | 28 |
| ID and SlaveID Input Ports. | 28 |
| MergedID, MergedSlaveID1, and MergedSlaveID2 Output Ports. | 28 |
| Rules and Guidelines for the Salesforce Merge Transformation. | 28 |
| Creating a Salesforce Merge Transformation. | 29 |
| Chapter 6: Salesforce PickList Transformation. | 30 |
| Salesforce PickList Transformation Overview. | 30 |
| Salesforce PickList Components. | 30 |
| Salesforce PickList Ports. | 31 |
| Rules and Guidelines for the Salesforce PickList Transformation. | 31 |
| Creating a Salesforce PickList Transformation. | 32 |
| Chapter 7: Salesforce Sessions and Workflows. | 33 |
| Salesforce Sessions and Workflows Overview. | 33 |
| Salesforce Connections. | 33 |
| Configuring a Salesforce Connection. | 34 |
| Configuring a Session with a Salesforce Source. | 34 |
| Filtering Source Data. | 35 |
| Capturing Deleted and Archived Salesforce Records. | 36 |
| Capturing Changed Data. | 36 |

| | |
|---|-----------|
| Continuous CDC Sessions. | 36 |
| Time-Period Based CDC Sessions. | 37 |
| Using the SystemModstamp or LastModifiedDate Timestamp for Change Data Capture. | 38 |
| Bulk API Source Sessions. | 38 |
| Configuring a Session with a Salesforce Target. | 39 |
| Configuring the Upsert Target Operation. | 40 |
| Configuring the Maximum Batch Size. | 41 |
| Handling Null Values in Update and Upsert Operations. | 41 |
| Logging PowerExchange for Salesforce Session Details. | 42 |
| Override an External ID with an idLookup for Upserts. | 43 |
| Bulk API Target Sessions. | 43 |
| Monitor a Bulk API Target Session. | 44 |
| Bulk API Target Load Type. | 44 |
| Bulk API Target Success Files and Error Files. | 44 |
| Hard Deletes with Bulk API Targets. | 47 |
| Configuring a Session for Optimal Performance. | 47 |
| Tuning the DTM Buffer Size. | 47 |
| Modifying the Precision of String Fields. | 48 |
| Appendix A: Datatype Reference. | 49 |
| Datatype Reference Overview. | 49 |
| Salesforce and Transformation Datatypes. | 49 |
| Appendix B: Glossary. | 52 |
| Index. | 54 |

Preface

The *PowerExchange for Salesforce User Guide for PowerCenter* provides information to build Salesforce mappings, extract data from Salesforce objects, and load data into Salesforce objects. It is written for the developers who are responsible for extracting data from Salesforce objects and loading data into Salesforce objects.

This book assumes you have knowledge of web services concepts, relational database concepts, PowerCenter, and Salesforce. You must also be familiar with the interface requirements for other supporting applications. For additional information about related Salesforce issues, refer to the Salesforce documentation.

Informatica Resources

Informatica My Support Portal

As an Informatica customer, you can access the Informatica My Support Portal at <http://mysupport.informatica.com>.

The site contains product information, user group information, newsletters, access to the Informatica customer support case management system (ATLAS), the Informatica How-To Library, the Informatica Knowledge Base, Informatica Product Documentation, and access to the Informatica user community.

Informatica Documentation

The Informatica Documentation team takes every effort to create accurate, usable documentation. If you have questions, comments, or ideas about this documentation, contact the Informatica Documentation team through email at infa_documentation@informatica.com. We will use your feedback to improve our documentation. Let us know if we can contact you regarding your comments.

The Documentation team updates documentation as needed. To get the latest documentation for your product, navigate to Product Documentation from <http://mysupport.informatica.com>.

Informatica Web Site

You can access the Informatica corporate web site at <http://www.informatica.com>. The site contains information about Informatica, its background, upcoming events, and sales offices. You will also find product and partner information. The services area of the site includes important information about technical support, training and education, and implementation services.

Informatica How-To Library

As an Informatica customer, you can access the Informatica How-To Library at <http://mysupport.informatica.com>. The How-To Library is a collection of resources to help you learn more about Informatica products and features. It includes articles and interactive demonstrations that provide solutions to common problems, compare features and behaviors, and guide you through performing specific real-world tasks.

Informatica Knowledge Base

As an Informatica customer, you can access the Informatica Knowledge Base at <http://mysupport.informatica.com>. Use the Knowledge Base to search for documented solutions to known technical issues about Informatica products. You can also find answers to frequently asked questions, technical white papers, and technical tips. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team through email at KB_Feedback@informatica.com.

Informatica Support YouTube Channel

You can access the Informatica Support YouTube channel at <http://www.youtube.com/user/INFASupport>. The Informatica Support YouTube channel includes videos about solutions that guide you through performing specific tasks. If you have questions, comments, or ideas about the Informatica Support YouTube channel, contact the Support YouTube team through email at supportvideos@informatica.com or send a tweet to @INFASupport.

Informatica Marketplace

The Informatica Marketplace is a forum where developers and partners can share solutions that augment, extend, or enhance data integration implementations. By leveraging any of the hundreds of solutions available on the Marketplace, you can improve your productivity and speed up time to implementation on your projects. You can access Informatica Marketplace at <http://www.informaticamarketplace.com>.

Informatica Velocity

You can access Informatica Velocity at <http://mysupport.informatica.com>. Developed from the real-world experience of hundreds of data management projects, Informatica Velocity represents the collective knowledge of our consultants who have worked with organizations from around the world to plan, develop, deploy, and maintain successful data management solutions. If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.

Informatica Global Customer Support

You can contact a Customer Support Center by telephone or through the Online Support.

Online Support requires a user name and password. You can request a user name and password at <http://mysupport.informatica.com>.

The telephone numbers for Informatica Global Customer Support are available from the Informatica web site at <http://www.informatica.com/us/services-and-training/support-services/global-support-centers/>.

CHAPTER 1

Understanding PowerExchange for Salesforce

This chapter includes the following topics:

- [Understanding PowerExchange for Salesforce Overview, 1](#)
- [PowerCenter and Salesforce Integration, 2](#)

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 Secure Sockets Layer (SSL) to encrypt all data that crosses the Internet. It uses 128-bit SSL encryption. PowerExchange for Salesforce also uses 128-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. 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

Installation and Configuration

This chapter includes the following topics:

- [Installation and Configuration Overview, 4](#)
- [Plug-in Registration, 5](#)
- [HTTP Proxy Options, 5](#)
- [Java Requirements for Bulk API Target Sessions, 7](#)
- [Uninstalling PowerExchange for Salesforce, 7](#)
- [Release Information, 7](#)

Installation and Configuration Overview

This chapter provides information about upgrading, installing, and configuring PowerExchange for Salesforce.

Prerequisites

Before you upgrade or install PowerExchange for Salesforce, install or upgrade PowerCenter on the machine that runs the Integration Service.

Installing and Configuring PowerExchange for Salesforce

To install and configure PowerExchange for Salesforce, complete the following steps:

1. Register the plug-in.
2. Configure the HTTP proxy options. Configure the HTTP proxy options for the PowerCenter Client and the PowerCenter Integration Service.

After you install and configure PowerExchange for Salesforce, you can create connections to access Salesforce. Create connection objects in the Workflow Manager so the PowerCenter Integration Service can connect to Salesforce.

Upgrading PowerExchange for Salesforce

When you upgrade PowerExchange for Salesforce, complete the following tasks:

1. Upgrade the plug-in registration.
2. Complete post-upgrade tasks.

Plug-in Registration

After you install PowerExchange for Salesforce, register the plug-in with the repository. If you are upgrading from a previous version, update the plug-in registration when you register the plug-in.

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 Informatica Administrator or 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 pmsfdc.xml to the following directory:

```
<PowerCenter 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.

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>\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>\client\bin\
```

Or, in the <PowerCenter Installation Directory>\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. |

| Parameter | Required/ Optional | Description |
|---------------|-----------------------|--|
| ProxyPassword | Required | Password for the proxy account. |
| Encrypted | Optional | <p>Use when the proxy password is encrypted. The proxy password should be encrypted using the PowerCenter command line program, pmpasswd.</p> <p>Use one of the following options:</p> <ul style="list-style-type: none"> - DEFAULT. Use for a password encrypted using the CRYPT_DATA encryption type. - SYSTEM. Use for a password encrypted using the CRYPT_SYSTEM encryption type. <p>Required if the proxy password is encrypted.</p> |

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. |

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.

Uninstalling PowerExchange for Salesforce

You can uninstall PowerExchange for Salesforce when you uninstall PowerCenter. For more information about uninstalling PowerCenter, see the *PowerCenter Installation Guide*.

Release Information

PowerExchange for Salesforce includes new features and changed behavior.

New Features

PowerExchange for Salesforce includes new features.

Salesforce Bulk API Source Sessions

Effective in 9.5.1.4.1, you can configure a session to use the Salesforce Bulk API to read data in bulk from a Salesforce source. By default, PowerExchange for Salesforce uses the standard Salesforce API.

Disassociating a Child Object

Effective in 9.5.1.4.1, you can disassociate a custom child object from a standard parent object. You can disassociate the object by performing a null update to the child custom object with the standard API.

CDC Flush Interval Offset

Effective in versions 9.5.1.2.1 and 9.5.1 HotFix 3, you can configure a CDC 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.

Prefix for Success and Error Log File Names

Effective in version 9.5.1 HotFix 1, you can add a prefix to success log file and error log file names.

Encrypted HTTP Proxy Account Password

Effective in version 9.5.0, you can use an encrypted HTTP proxy account password. Use the PowerCenter command line program, `pmpasswd`, to encrypt the password.

Set Fields to NULL with the Bulk API

Effective in version 9.1.0 HotFix 3, you can configure a Bulk API target session to replace values in the target with null values from the source. Use the existing Set Fields to NULL session property when configuring a Bulk API target session.

Bulk API Success and Error Logs

Effective in version 9.1.0 HotFix 2, you can generate success and error logs separately for Bulk API target sessions. When you monitor a Bulk API target session, you can use the existing Use SFDC Success File session property to generate a success file and the Use SFDC Error File session property to generate an error file.

Truncate Overflow Data for Salesforce Targets

Effective in version 9.1.0, you can configure a session to have Salesforce truncate overflow target data and write truncated data to a Salesforce target. By default, the PowerCenter Integration Service writes overflow data to the session error file.

Forced Authentication HTTP Proxy Servers

Effective in version 9.1.0, PowerExchange for Salesforce supports forced authentication HTTP proxy servers.

Salesforce Bulk API Target Sessions

Effective in version 9.0.1, you can use the Salesforce Bulk API to load bulk data into a Salesforce target. Salesforce Bulk API can insert, update, upsert, and delete records. In a Bulk API target session, you can

perform parallel or serial loads, log information about batch processing in the session log, and create success and error logs. By default, PowerExchange for Salesforce uses the standard Salesforce API.

Viewing Record-Level Session Logs

Effective in version 8.6.0.1, the PowerCenter Integration Service generates record-level logs for each session that writes to a Salesforce target. The PowerCenter Integration Service generates the following types of logs:

- **Success log.** The success log contains an entry for each record that is successfully inserted, updated, deleted, or upserted. Each entry contains the values loaded for all fields of the record. Use this file to understand what data is loaded into the Salesforce target.
- **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. Use this file to understand why records did not load into the Salesforce target.

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

You can configure whether the PowerCenter Integration Service generates the success and error logs and the directories where the logs are stored.

Capturing Deleted and Archived Salesforce Records

Effective in version 8.6.0.1, the PowerCenter Integration Service can capture deleted and archived records from a Salesforce source object. By default, mappings do not capture deleted or archived records.

Supporting Salesforce Professional Edition Accounts

Effective in version 8.6.0.1, the PowerCenter Integration Service can extract data from and load data into Salesforce Professional edition accounts.

Merging Duplicate Salesforce Records

Effective in version 8.6.0.1, PowerExchange for Salesforce contains the Salesforce Merge transformation. Use the Salesforce Merge transformation to merge duplicate records into a single record.

Validating Salesforce PickList Values

Effective in version 8.6.0.1, PowerExchange for Salesforce contains the Salesforce PickList transformation. Use the Salesforce PickList transformation to retrieve picklist values from Salesforce and store them in the PowerCenter repository. Use the picklist values to validate values that are loaded into a Salesforce target.

Changed Behavior

PowerExchange for Salesforce includes changed behavior.

Java Requirement for Bulk API Target Sessions

Effective in version 9.5.1.4.1, for Bulk API target sessions, configure at least 10 to 50 MB of space for the Java temporary directory on the PowerCenter Integration Service machine.

Previously, the Bulk API did not use the Java temporary directory when writing to Salesforce targets.

End of Life for Salesforce API Versions

Effective in version 9.5.1.4.1, PowerExchange for Salesforce does not support the following Salesforce API versions:

- 7.0
- 8.0
- 16.0

Previously, PowerExchange for Salesforce supported these Salesforce API versions.

Related Salesforce Object Fields Not Available for Import

Effective in version 9.5.1.4.1, you cannot import fields from objects related to the following Salesforce objects:

- ActivityHistory
- EmailStatus
- Name
- OpenActivity
- OwnedContentDocument

Previously, you could import fields from objects related to these Salesforce objects.

For more information, see [“Rules and Guidelines for Importing Fields from Related Salesforce Objects” on page 15](#).

Error Logging

Effective in 9.5.1.4.1, the PowerCenter Integration Service writes error messages to the error log for the session.

Previously, the PowerCenter Integration Service wrote error messages to both the error log and the session log.

SOAP Request Logging

Effective in version 9.5.1.4.1, for sessions that read from Salesforce with the standard API, the PowerCenter Integration Service no longer includes SOAP requests in the session log.

Previously, you could view SOAP requests in session logs when you configured the session for verbose tracing.

Salesforce API Version

Effective in version 9.5.1.4.1, PowerExchange for Salesforce uses version 30.0 of the Salesforce API.

Use the Salesforce service URL to configure connections to Salesforce. To use the latest version of the Salesforce API, create an application connection or update the service URL in an existing application connection.

Use the following version of the Salesforce service URL:

```
https://www.salesforce.com/services/Soap/u/30.0
```

If the new version of a Salesforce object has a different structure than the previous version of the object, re-import the Salesforce object. After you re-import the object, analyze the associated mapping to determine if you need to update the mapping.

Note: You cannot import the following Salesforce objects that are newly available with version 30.0 of the Salesforce API:

- AttachedContentDocument
- CombinedAttachment

Previously, PowerExchange for Salesforce used version 27.0 of the Salesforce API.

The following table lists the default service URLs for earlier versions of PowerExchange for Salesforce:

| PowerExchange for Salesforce Version | Salesforce Service URL |
|---|---|
| 9.5.1 HotFix 1 | https://www.salesforce.com/services/Soap/u/27.0 |
| 9.5.1 | https://www.salesforce.com/services/Soap/u/26.0 |
| 9.5.0 | https://www.salesforce.com/services/Soap/u/24.0 |
| 9.1.0 HotFix 5 | https://www.salesforce.com/services/Soap/u/25.0 |
| 9.1.0 HotFix 3, 9.1.0 HotFix 4 | https://www.salesforce.com/services/Soap/u/23.0 |
| 9.1.0 HotFix 2 | https://www.salesforce.com/services/Soap/u/21.0 |
| 9.1.0, 9.1.0 HotFix 1 | https://www.salesforce.com/services/Soap/u/20.0 |
| 9.0.1 HotFix 1, 9.0.1 HotFix 2, and 8.6.1.0.4 | https://www.salesforce.com/services/Soap/u/19.0 |
| 9.0.1 | https://www.salesforce.com/services/Soap/u/18.0 |
| 9.0 | https://www.salesforce.com/services/Soap/u/17.0 |

Generating Bulk API Success and Error Files

Effective in version 9.1.0 HotFix 2, success and error logs are no longer generated by default when you monitor a Bulk API target session.

To generate success and error logs for Bulk API target sessions, select the Use SFDC Success File or Use SFDC Error File session properties, and select the Monitor Bulk Job Until All Batches Processed session property to monitor the session.

Previously, success and error logs were generated by default when you monitored a Bulk API target session, and you could not generate the logs separately.

Converting Time Data

Effective in version 9.0.1, PowerExchange for Salesforce trims the date when it converts the PowerCenter Date/Time transformation datatype to the Salesforce time datatype.

Previously, when you configured a mapping to write to the Salesforce time datatype, you had to trim the date from the Date/Time transformation datatype. Existing mappings are unaffected by this change.

Specifying the External ID or idLookup Field for an Upsert Operation

Effective in version 8.6.1 HotFix 5, you can specify the external ID or idLookup field to use for an upsert operation on a Salesforce target.

Previously, when you configured a session to upsert records into a Salesforce target and the Salesforce target had multiple external ID or idLookup fields, the PowerCenter Integration Service used the first external ID or idLookup field to identify existing target records.

Updating the Repository Plug-in

Effective in version 8.6.1, you must update the repository plug-in.

RELATED TOPICS:

- [“Updating the Repository Plug-in” on page 12](#)

Configuring Dates for Change Data Capture

Effective in version 8.6.1, you can use the SystemModstamp or LastModifiedDate field for a time-period based change data capture session. Default is LastModifiedDate. When you upgrade to version 8.6.1, existing sessions continue to use the LastModifiedDate field. To increase session performance, configure sessions to use the SystemModstamp field. Salesforce indexes the SystemModstamp field, not the LastModifiedDate field.

Previously, all sessions used the LastModifiedDate field.

Upserting Records Based on a Salesforce idLookup Field

Effective in version 8.6.1, you can use the Salesforce idLookup or external ID fields to upsert records into a Salesforce object.

Previously, you had to use the external ID to upsert records.

Converting Datatypes

Effective in version 8.6.1, PowerExchange for Salesforce supports the following datatype conversions:

| Salesforce Datatype | Range and Description | PowerCenter Datatype | Range and Description |
|---------------------|--|----------------------|--|
| 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 |
| time | Time values. | Date/Time | Precision 1 to 28 digits, scale 0 to 28. PowerCenter Integration Service adds the current date to the time when converting from the time Salesforce datatype to the Date/Time transformation datatype. |

Previously, PowerExchange for Salesforce did not support the Salesforce encryptedString or Time datatypes.

Handling Null Values in Update and Upsert Operations

Effective in version 8.6.0.1, you can configure the PowerCenter Integration Service to replace existing values in a Salesforce record with null values from a source during an update or upsert operation. By default, the PowerCenter Integration Service does not replace existing values with null values.

Previously, the PowerCenter Integration Service replaced existing values with null values by default. When you upgrade, all sessions are not configured to replace existing values with null values. To update existing value with null values, configure the Set Fields to NULL session property.

Upserting Related Object Records

Effective in version 8.6.0.1, the PowerCenter Integration Service can upsert records from more than two related objects into a Salesforce target object.

Previously, you could upsert records from one or two related objects.

CHAPTER 3

Salesforce Sources and Targets

This chapter includes the following topics:

- [Salesforce Sources and Targets Overview, 14](#)
- [Importing Fields from Related Salesforce Objects, 14](#)
- [Salesforce Sources, 16](#)
- [Salesforce Targets, 16](#)
- [Importing a Salesforce Source or Target Definition, 18](#)

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.

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 if the related object is a parent object.
- You cannot import fields from a related object if the Salesforce definition is based one of the following Salesforce objects:
 - ActivityHistory
 - Attachment
 - EmailStatus
 - Name
 - Note
 - NoteAndAttachment
 - OpenActivity
 - OwnedContentDocument
 - ProcessInstance
 - ProcessInstanceHistory

- 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 when you want to look up data in the primary Salesforce object and related objects and do not want to perform a join. For example, you can lookup 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 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 41](#)

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 it is defined in the Salesforce target definition. |
| Createable | Indicates whether the field allows inserts: <ul style="list-style-type: none">- 0. False, the PowerCenter Integration Service cannot insert data in this field.- 1. True, the PowerCenter Integration Service can insert data in this field. A PowerCenter insert is equivalent to a Salesforce create operation. |
| Updateable | Indicates whether the field allows updates: <ul style="list-style-type: none">- 0. False, the PowerCenter Integration Service cannot update data in this field.- 1. True, the PowerCenter Integration Service can update data in this field. |
| External ID | Salesforce custom fields only. Indicates whether the field is designated as an external ID field: <ul style="list-style-type: none">- 0. False, the field is not designated as an external ID field.- 1. True, the field is designated as an external ID field. Each Salesforce object can contain a single custom field designated as the external ID field. Salesforce appends custom field names with “__c”. For more information about external ID and custom fields, 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 also insert, update, and upsert records in a Salesforce target 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 Tables from Salesforce dialog box, enter the following information:

| Import Attribute | Description |
|------------------|--|
| User Name | Salesforce user name. |
| Password | Password for the Salesforce user name. Password is case sensitive. |
| Service URL | URL of the Salesforce service you want to access. 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. |

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, 20](#)
- [Salesforce Lookup Components, 21](#)
- [Salesforce Lookup Ports, 22](#)
- [Salesforce Lookup Query, 24](#)
- [Creating a Salesforce Lookup Transformation, 24](#)

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.

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.

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 24](#)
- [“Salesforce and Transformation Datatypes” on page 49](#)

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 24](#)

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>'
```

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 that you want to access. 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, 26](#)
- [Salesforce Merge Components, 27](#)
- [Salesforce Merge Ports, 28](#)
- [Rules and Guidelines for the Salesforce Merge Transformation, 28](#)
- [Creating a Salesforce Merge Transformation, 29](#)

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 |
|--------------------|--------------|--------------|--------------|
| 0015000000lcEgAAAV | ABC Tiles | Los Angeles | - |
| 0015000000lcEgBAAV | ABC Tiles | - | 310-555-1212 |
| 0015000000lcEgCAAV | 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 |
|--------------------|--------------------|--------------------|----------------|
| 0015000000lcEgAAAV | 0015000000lcEgBAAV | 0015000000lcEgCAAV | 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 |
|--------------------|----------------|--------------|-------|
| 0015000000lcEgAAAV | Textiles to Go | Los Angeles | - |

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 49](#)

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:

- 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 that you want to access. 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, 30](#)
- [Salesforce PickList Components, 30](#)
- [Salesforce PickList Ports, 31](#)
- [Rules and Guidelines for the Salesforce PickList Transformation, 31](#)
- [Creating a Salesforce PickList Transformation, 32](#)

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:

- 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 that you want to access. 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, 33](#)
- [Salesforce Connections, 33](#)
- [Configuring a Session with a Salesforce Source, 34](#)
- [Configuring a Session with a Salesforce Target, 39](#)
- [Configuring a Session for Optimal Performance, 47](#)

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 run-time connection.

The PowerCenter Integration Service uses the Salesforce security mechanism to authenticate the login. If you provide an invalid login, 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 obtaining a valid Salesforce login, see the Salesforce administrator for your organization.

Note: You cannot use a connection variable for an application connection.

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.
5. In the Connection Object Definition dialog box, enter a name for the application connection.
6. Enter the Salesforce user name for the application connection.
The PowerCenter Integration Service uses this user name to log in to Salesforce.
7. Enter the password for the Salesforce user name.
8. Enter the Salesforce service URL.
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.
9. Click OK.
The new application connection appears in the Application Object Browser.

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. |

| Property Name | Description |
|----------------------------|---|
| 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. |
| 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. |
| 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. |

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 read from the source. When you enter a filter condition, the PowerCenter Integration Service modifies and overrides the WHERE clause in the SOQL query.

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'
```

Enter a filter condition based on the SOQL syntax in the Salesforce documentation. 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 a LIMIT clause in the filter condition.
- You enter an AND or OR operator in the filter condition, but you do not enclose the AND or OR statement in parentheses. Enclose the AND or OR statement in parentheses as shown in the following example:

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

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.

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

Capturing Changed Data

The PowerCenter Integration Service can capture changed data from a Salesforce object that is replicatable and contains CreatedDate and LastModifiedDate fields. If you configure a session to capture changed data from a Salesforce object that is not replicatable or does not contain CreatedDate and LastModifiedDate fields, the session fails. For more information about replicatable 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. If you configure properties for both methods, you may encounter unexpected results.

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 38](#)

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 Salesforce 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.

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 Updates | The exact name of the external ID or idLookup field to use for updates. 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 updates. |

| Property Name | Description |
|--|--|
| 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. |
| 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. |

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 47](#)

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 to 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. When the PowerCenter Integration Service creates a batch, it adds required characters to properly format the data, such as adding quotation marks around text.

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.

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><batch id>_success.csv |
| Bulk API error log | \$PMBadFileDir | <session name><timestamp><batch id>_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><batch id>_success.csv |
| Bulk API error log | \$PMBadFileDir | <prefix>_<session name><timestamp><batch id>_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. |
| Success | Indicates the row was successfully written to the target. For all rows in the Bulk API success log, default is True. |

| Column Name | Description |
|-------------|--|
| 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 | IDS | Success | Created | Error | ID | Name |
|------------------------|----------|----------|---------|---------|--|--------------|-----------|
| 750i000000751i0000000- | 13Q7SAAU | 1Jzz7AAC | FALSE | FALSE | MALFORMED_ID:Account ID: id value of incorrect type: QHP1NÄfRAA Electric | Ä`Ä»Ä¿ | MDM Water |
| 750i000000751i0000000- | 13Q7SAAU | 1Jzz7AAC | FALSE | FALSE | MALFORMED_ID:Account ID: id value of incorrect type: QHP1NÄfRAA Electric | ÄfÄ`Ä'Ä»Ä'Ä¿ | MDM Water |

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, 49](#)
- [Salesforce and Transformation Datatypes, 49](#)

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 |
| 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 |

| Salesforce Datatype | Range and Description | Transformation Datatype | Range and Description |
|---------------------|--|-------------------------|---|
| 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 |
| 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 replicatable 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.

replicatable object

A Salesforce object that can be duplicated in another environment. Every Salesforce object has a replicatable 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

A

- application connections
 - configuring for Salesforce [34](#)
 - description for Salesforce [2](#)
 - endpoint URL for Salesforce [33](#)
 - Salesforce [33](#)
 - Salesforce API version [10](#)
- authentication
 - user authentication for Salesforce [2](#), [33](#)

B

- batch size
 - description for Salesforce [41](#)
- Bulk API
 - error log [46](#)
 - hard deletes from Salesforce targets [47](#)
 - Java temporary directory requirements [7](#)
 - success and error logs [8](#)
 - success log [45](#)
 - success logs and error logs for targets [44](#)
- Bulk API source session
 - configuring for Salesforce [38](#)
- Bulk API success and error files
 - behavior updates [11](#)
- bulk API target session
 - configuring for Salesforce [39](#)
- Bulk API target session
 - available for Salesforce [8](#)
 - configuring for Salesforce [43](#)
 - configuring the load type for Salesforce [44](#)
 - monitoring for Salesforce [44](#)

C

- CDC End Timestamp
 - configuring for Salesforce [34](#)
- CDC Flush Interval Offset
 - description for Salesforce [37](#)
- CDC Start Timestamp
 - configuring for Salesforce [34](#)
- CDC Time Limit
 - configuring for Salesforce [34](#)
 - description for Salesforce [36](#)
- change data capture
 - configuring for Salesforce [36](#)
 - continuous capture for Salesforce [36](#), [37](#)
 - rules for continuous capture for Salesforce [36](#)
 - rules for time-period capture for Salesforce [37](#), [38](#)
 - time-period capture for Salesforce [37](#), [38](#)
 - timestamp configuration for Salesforce [38](#)
- code pages
 - supported code pages for Salesforce [3](#)

- code pages (*continued*)
 - validation for Salesforce [3](#)
- configuration
 - PowerExchange for Salesforce [4](#)
- configuring for Salesforce
 - Bulk API source sessions [38](#)
 - Bulk API target sessions [43](#)
- connection objects
 - description for Salesforce [2](#)
- continuous workflows
 - and change data capture for Salesforce [36](#)
- Createable attribute
 - description for Salesforce [17](#)
- custom fields
 - external ID attribute for Salesforce [14](#)
- custom objects
 - description for Salesforce [2](#)

D

- data access controls
 - description for Salesforce [1](#)
- data movement mode
 - ASCII for Salesforce [3](#)
 - Unicode for Salesforce [3](#)
- datatypes
 - conversion for Salesforce [16](#), [49](#)
 - Salesforce [49](#)
- descendant object
 - description for Salesforce [14](#)
- Designer
 - integration with Salesforce [2](#)
- Disable Bulk Success and Error File Creation
 - configuring for Salesforce [39](#)
- disassociate
 - a custom child object with the upsert operation [42](#)
- DTM Buffer Size
 - configuring for Salesforce [41](#)
 - tuning for Salesforce sessions [47](#)

E

- Enable Bulk Query
 - configuring for Salesforce sources [34](#)
- Enable Field Truncation Attribute
 - configuring for Salesforce [39](#)
- Enable Hard Deletes for Bulk API
 - configuring for Salesforce [39](#)
- encrypting
 - HTTP proxy password [6](#)
- error logs
 - for Salesforce Bulk API target sessions [44](#)
 - Salesforce Bulk API [46](#)

- external ID
 - description [40](#)
 - description for Salesforce [16](#)
 - overriding with idLookup [43](#)
- external ID attribute
 - description for Salesforce [17](#)

F

- field attributes
 - description for Salesforce [17](#)
- filter conditions
 - on Salesforce lookup [23](#)
 - Salesforce data [35](#)
- filters, for Salesforce source [35](#)
- Flush Interval
 - configuring for Salesforce [34](#)
 - description for Salesforce [36](#)
- forced authentication
 - HTTP proxy server [8](#)

H

- HTTP proxy options
 - configuring [5](#)
 - configuring for PowerCenter Client (Salesforce) [5](#)
 - configuring for PowerCenter Integration Service (Salesforce) [6](#)
- HTTP proxy server
 - forced authentication support [8](#)

I

- idLookup
 - description for Salesforce [40](#)
 - overriding the external ID [43](#)
- IDs
 - external and Salesforce [16](#)
- installation
 - PowerExchange for Salesforce [4](#)

J

- Java
 - requirements for Bulk API target sessions [7](#)

L

- LastModifiedDate
 - description for Salesforce [38](#)
- LKP_FILTER port
 - description for Salesforce [23](#)
- LKP_MATCHIDX port
 - description for Salesforce [23](#)
- lookup ports
 - in Salesforce Lookup transformation [22](#)
- lookup query
 - description for Salesforce [24](#)
 - Salesforce matches [23](#)
- Lookup transformations
 - importing fields from related Salesforce objects [14](#), [15](#)

M

- mapping variables
 - for CDC in Salesforce sessions [34](#)
- Max Batch Size
 - configuring for Salesforce [39](#)
- Merge transformations
 - importing fields from related Salesforce objects [14](#), [15](#)
- Monitor Bulk Job Until All Batches Processed
 - configuring for Salesforce [39](#)
- monitoring
 - Salesforce Bulk API target sessions [44](#)

N

- naming conventions
 - for related Salesforce objects [14](#)
- nulls
 - handling in upserts and updates [41](#)

O

- object types
 - Salesforce standard and custom [2](#)
- output ports
 - in Salesforce PickList transformation [31](#)
- Override Parallel Concurrency with Serial
 - configuring for Salesforce [39](#)

P

- \$Paramend
 - mapping variable [34](#)
- \$Paramstart
 - mapping variable [34](#)
- pass-through ports
 - in Salesforce Lookup transformation [23](#)
 - in Salesforce PickList transformation [31](#)
- performance
 - configuring buffer block size for Salesforce [41](#)
 - configuring DTM Buffer Size for Salesforce [47](#)
- permissions
 - on Salesforce data [1](#)
- Picklist transformations
 - importing fields from related Salesforce objects [14](#), [15](#)
- PickList_Input port
 - in Salesforce PickList transformation [31](#)
- plug-ins
 - registration for PowerExchange for Salesforce [5](#)
- pmsfdc.ini
 - PowerExchange for Salesforce configuration file [5](#)
- pmsfdc.xml
 - Salesforce plug-in [5](#)
- ports
 - types in Salesforce Lookup transformation [22](#)
 - types in Salesforce Merge transformation [28](#)
 - types in Salesforce PickList transformation [31](#)
- PowerCenter
 - integration with Salesforce [1](#)
- PowerCenter Integration Service
 - integration with Salesforce [2](#)
- PowerExchange for Salesforce
 - application connection API version [10](#)
 - changed behavior [9](#)
 - overview [1](#)

- PowerExchange for Salesforce (*continued*)
 - post-upgrade tasks [9](#)
- precision
 - modifying for Salesforce fields [48](#)
- prerequisites
 - PowerExchange for Salesforce [4](#)
- primary Salesforce object
 - description [14](#)
- proxy password
 - encrypting [6](#)

R

- related objects
 - importing from Salesforce [14](#)
- related objects:
 - importing fields from Salesforce [15](#)
- relationships
 - importing related object fields from Salesforce [14](#)
- replicableable object
 - description [36](#)
- repository plug-in
 - updating for Salesforce [12](#)
- Row Limit
 - configuring for Salesforce [34](#)
- rules
 - configuring update strategy for Salesforce targets [17](#)
 - for continuous change data capture for Salesforce [36](#)
 - for importing fields from related Salesforce objects [15](#)
 - for time-period based change data capture for Salesforce [38](#)

S

- Salesforce
 - datatypes [49](#)
- Salesforce API version
 - used in the application connection [10](#)
- Salesforce Bulk API
 - targets [8](#)
- Salesforce definitions
 - overview [14](#)
- Salesforce Lookup transformation
 - compared to Lookup transformation [20](#)
 - components [21](#)
 - creating [24](#)
 - default ports [22](#)
 - lookup filter [23](#)
 - lookup ports [22](#)
 - matches, identifying [23](#)
 - overview [20](#)
 - pass-through ports [23](#)
 - port types [22](#)
- Salesforce Merge transformation
 - components [27](#)
 - creating [29](#)
 - default ports [28](#)
 - guidelines [28](#)
 - overview [26](#)
 - port types [28](#)
- Salesforce objects
 - attributes [17](#)
 - description [1](#)
 - permissions [1](#)
- Salesforce PickList transformation
 - components [30](#)
 - creating [32](#)

- Salesforce PickList transformation (*continued*)
 - guidelines [31](#)
 - output ports [31](#)
 - overview [30](#)
 - pass-through ports [31](#)
 - PickList_Input port [31](#)
 - port types [31](#)
- Salesforce Sandbox
 - accessing [18](#)
- Salesforce session
 - logging session details [42](#)
- Salesforce sessions
 - continuous change data capture [36](#)
 - time-period change data capture [37](#)
- Salesforce source definitions
 - creating [16](#)
- Salesforce sources
 - time zone handling [16](#)
- Salesforce targets
 - hard deletes with the Bulk API [47](#)
 - time zone requirement [17](#)
- session conditions
 - DTM Buffer Size for Salesforce [41](#)
 - Row Limit for Salesforce [34](#)
- session details
 - for Salesforce sessions [42](#)
- session properties
 - for Salesforce [39](#)
- session property
 - CDC End Timestamp for Salesforce [34](#)
 - CDC Start Timestamp for Salesforce [34](#)
 - CDC Time Limit for Salesforce [34](#)
 - Flush Interval for Salesforce [34](#)
 - SOQL Filter Condition for Salesforce [34](#)
 - Use queryAll for Salesforce [34](#)
 - Use SystemModstamp for CDC for Salesforce [34](#)
- sessions
 - overview [33](#)
 - tuning DTM Buffer Size for Salesforce [47](#)
 - tuning joins for Salesforce [15](#)
- Set Fields to NULL
 - configuring for Salesforce [39](#)
 - session property [41](#)
- Set Prefix for Error Files
 - configuring for Salesforce [39](#)
- Set Prefix for Success Files
 - configuring for Salesforce [39](#)
- Set the Interval for Polling Bulk Job Status
 - configuring for Salesforce [39](#)
- Set the Location of the Bulk Error Files
 - configuring for Salesforce [39](#)
- SFDC Success File Directory
 - configuring for Salesforce [39](#)
- Sforce Object Query Language (SOQL)
 - description [2](#)
- SOQL Filter Condition
 - configuring for Salesforce [35](#)
- SOQL override
 - configuring [35](#)
 - description [35](#)
- source definition
 - importing from Salesforce [18](#)
- source definitions
 - overview for Salesforce [14](#)
- sources
 - importing fields from related Salesforce objects [14](#), [15](#)
 - Salesforce data, filtering [35](#)

SSL

transport-layer security for Salesforce [2](#)

success logs

for Salesforce Bulk API target sessions [44](#)

Salesforce Bulk API [45](#)

SystemModstamp

description [38](#)

T

target definition

importing from Salesforce [18](#)

target definitions

overview for Salesforce [14](#)

target loading

Salesforce Bulk API target sessions [44](#)

target operations

PowerExchange for Salesforce [16](#)

targets

importing fields from related Salesforce objects [14](#), [15](#)

time zone handling

for Salesforce sources [16](#)

time zone requirement

for Salesforce targets [17](#)

timestamp

configuring for change data capture for Salesforce [38](#)

transformations

Salesforce Lookup [20](#)

Salesforce Merge [26](#)

Salesforce PickList [30](#)

Treat Insert as Upsert

configuring for Salesforce [39](#)

Treat Update as Upsert

configuring for Salesforce [39](#)

U

uninstalling

PowerExchange for Salesforce [7](#)

Updateable attribute

description for Salesforce [17](#)

upgrading

PowerExchange for Salesforce upgrade tasks [4](#)

upsert

configuring for Salesforce [40](#)

description for Salesforce [40](#)

disassociate a custom child object [42](#)

external ID [40](#)

overriding external ID with idLookup [43](#)

Salesforce idLookup field [40](#)

Salesforce session configuration [40](#)

Use Idlookup Field for Upserts

configuring for Salesforce [39](#)

Use queryAll

configuring for Salesforce [34](#)

Use SFDC Bulk API

configuring for Salesforce targets [39](#)

Use SFDC Error File

configuring for Salesforce [39](#)

Use SFDC Success File

configuring for Salesforce [39](#)

Use SystemModstamp for CDC

configuring for Salesforce [34](#)

Use this ExternalId/IdLookup field for Updates

configuring for Salesforce [39](#)

V

validation

SOQL syntax [2](#)