



Informatica® PowerExchange for HDFS
10.0

User Guide

© Copyright Informatica LLC 2012, 2018

This software and documentation contain proprietary information of Informatica LLC 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 LLC. 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(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 LLC 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 © 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. Copyright © Scalable Systems, Inc. All rights reserved. Copyright © jqWidgets. All rights reserved. Copyright © Tableau Software, Inc. All rights reserved. Copyright © MaxMind, Inc. All Rights Reserved. Copyright © TMat Software s.r.o. All rights reserved. Copyright © MapR Technologies Inc. All rights reserved. Copyright © Amazon Corporate LLC. All rights reserved. Copyright © Highsoft. All rights reserved. Copyright © Python Software Foundation. All rights reserved. Copyright © BeOpen.com. All rights reserved. Copyright © CNRI. 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/IODBC/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.schneider.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>, <https://github.com/lyokato/libgeohash/blob/master/LICENSE>, <https://github.com/hjiang/jsonxx/blob/master/LICENSE>, <https://code.google.com/p/lz4/>, <https://github.com/jedisct1/libsodium/blob/master/LICENSE>, <http://one-jar.sourceforge.net/index.php?page=documents&file=license>, <https://github.com/EsotericSoftware/kryo/blob/master/license.txt>, <http://www.scala-lang.org/license.html>, <https://github.com/tinkerpop/blueprints/blob/master/LICENSE.txt>, <http://gee.cs.oswego.edu/dl/classes/EDU/oswego/cs/dl/util/concurrent/intro.html>, <https://aws.amazon.com/ssl/>, <https://github.com/twbs/bootstrap/blob/master/LICENSE>, <https://sourceforge.net/p/xmlunit/code/HEAD/tree/trunk/LICENSE.txt>, <https://github.com/documentcloud/underscore-contrib/blob/master/LICENSE>, and <https://github.com/apache/hbase/blob/master/LICENSE.txt>.

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.

See patents at <https://www.informatica.com/legal/patents.html>.

DISCLAIMER: Informatica LLC 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 LLC 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.

Publication Date: 2018-09-27

Table of Contents

Preface	6
Informatica Resources.	6
Informatica My Support Portal.	6
Informatica Documentation.	6
Informatica Product Availability Matrixes.	6
Informatica Web Site.	7
Informatica How-To Library.	7
Informatica Knowledge Base.	7
Informatica Support YouTube Channel.	7
Informatica Marketplace.	7
Informatica Velocity.	7
Informatica Global Customer Support.	7
 Chapter 1: Introduction to PowerExchange for HDFS.....	 9
PowerExchange for HDFS Overview.	9
 Chapter 2: PowerExchange for HDFS Configuration.....	 10
PowerExchange for HDFS Configuration Overview.	10
Prerequisites.	10
MapR Prerequisites.	11
Hortonworks Prerequisites.	12
 Chapter 3: HDFS Connections.....	 13
HDFS Connections Overview.	13
HDFS Connection Properties.	13
Creating an HDFS Connection.	14
 Chapter 4: HDFS Data Objects.....	 15
HDFS Data Objects Overview.	15
Flat File Data Objects.	15
Compression and Decompression for Flat File Sources and Targets.	16
Rules and Guidelines for Flat File Data Objects.	17
Configuring a Flat File Data Object with an HDFS Connection.	17
Complex File Data Objects.	17
Complex File Data Object Overview Properties.	18
Compression and Decompression for Complex File Sources and Targets.	18
Parameterization of Complex File Data Objects.	19
Creating a Complex File Data Object.	20
Complex File Data Object Output Parsing.	20
Custom Formats.	21

Custom Formats Configuration.	21
Chapter 5: HDFS Data Extraction.	22
HDFS Data Extraction Overview.	22
Flat File Data Object Read Properties.	22
Complex Files Partitioning.	23
Complex File Data Object Read Properties.	23
General Properties.	24
Ports Properties.	24
Sources Properties.	24
Advanced Properties.	25
Chapter 6: HDFS Data Load.	26
HDFS Data Load Overview.	26
Flat File Data Object Write Properties.	26
Complex File Streaming.	27
Complex Files Output Collection Mode.	28
Complex File Data Object Write Properties.	29
General Properties.	30
Port Properties.	30
Sources Properties.	30
Advanced Properties.	31
Chapter 7: HDFS Mappings.	33
HDFS Mappings Overview.	33
Hive Validation and Run-time Environments.	33
HDFS Data Extraction Mapping Example.	34
HDFS Data Load Mapping Example.	35
Appendix A: Data Type Reference.	37
Data Type Reference Overview.	37
Flat File and Transformation Data Types.	37
Complex File and Transformation Data Types.	38
Index.	39

Preface

The *Informatica PowerExchange for HDFS User Guide* provides information about reading data from the Hadoop Distributed File System (HDFS) and writing data to HDFS. The guide is written for database administrators and developers.

This book assumes you have knowledge of HDFS and Informatica Developer.

Informatica Resources

Informatica My Support Portal

As an Informatica customer, the first step in reaching out to Informatica is through the Informatica My Support Portal at <https://mysupport.informatica.com>. The My Support Portal is the largest online data integration collaboration platform with over 100,000 Informatica customers and partners worldwide.

As a member, you can:

- Access all of your Informatica resources in one place.
- Review your support cases.
- Search the Knowledge Base, find product documentation, access how-to documents, and watch support videos.
- Find your local Informatica User Group Network and collaborate with your peers.

Informatica Documentation

The Informatica Documentation team makes 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 <https://mysupport.informatica.com>.

Informatica Product Availability Matrixes

Product Availability Matrixes (PAMs) indicate the versions of operating systems, databases, and other types of data sources and targets that a product release supports. You can access the PAMs on the Informatica My Support Portal at <https://mysupport.informatica.com>.

Informatica Web Site

You can access the Informatica corporate web site at <https://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 <https://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 <https://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 <https://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

Introduction to PowerExchange for HDFS

This chapter includes the following topic:

- [PowerExchange for HDFS Overview, 9](#)

PowerExchange for HDFS Overview

PowerExchange for HDFS provides connectivity to the Hadoop Distributed File System (HDFS). You can use PowerExchange for HDFS to read data from and write data to HDFS. You can also use PowerExchange for HDFS to read data from and write data to the local file system.

The Data Integration Service uses the Hadoop API infrastructure to connect to HDFS. It connects to the NameNode to read data from and write data to HDFS.

With PowerExchange for HDFS, you can read and write fixed-width and delimited file formats with a flat file data object. You can also read and write compressed files.

You can read and write text files and binary file formats such as sequence files with a complex file data object. You can also read and write compressed files.

You can configure custom formats to process data in input, output, and compression formats that Hadoop supports.

CHAPTER 2

PowerExchange for HDFS Configuration

This chapter includes the following topics:

- [PowerExchange for HDFS Configuration Overview, 10](#)
- [Prerequisites, 10](#)
- [MapR Prerequisites, 11](#)
- [Hortonworks Prerequisites, 12](#)

PowerExchange for HDFS Configuration Overview

PowerExchange for HDFS is installed with Informatica Data Services. You enable PowerExchange for HDFS with a license key.

Note: To read or write data with a complex file data object, you will also need the Unstructured Data license key.

Prerequisites

Before you use PowerExchange for HDFS to access data in HDFS, perform the following tasks:

1. Install Informatica Data Services.
2. Install Big Data Management if you want to run mappings in a Hive run-time environment.
3. On the machine where you installed the Developer tool, verify that the Hadoop Distribution Directory property in the `developerCore.ini` file is set based on the Hadoop distribution that you use.

The `developerCore.ini` file is located in the following directory:

```
<Client installation directory>\clients\DeveloperClient\
```

The following table lists the Hadoop Distribution Directory property value that you must set for different Hadoop distributions:

Hadoop Distribution	Hadoop Distribution Directory Property Value
Cloudera	-DINFA_HADOOP_DIST_DIR=hadoop\cloudera_<version>
Hortonworks	-DINFA_HADOOP_DIST_DIR=hadoop\hortonworks_<version>
MapR	-DINFA_HADOOP_DIST_DIR=hadoop\mapr_<version>

Note: By default, the Hadoop Distribution Directory property value is set for a Cloudera distribution.

- In the Administrator tool, verify that the following distribution directory properties in the **Processes** tab of the **Data Integration Service** page are set based on the Hadoop distribution that you use:
 - Data Integration Service Hadoop Distribution Directory
 - Hadoop Distribution Directory
 For more information about the distribution directory properties, see *Big Data Management User Guide*.

MapR Prerequisites

If you use the MapR Hadoop distribution to access data in HDFS, perform the following additional tasks:

- In the Administrator tool, add the -Dmapr.library.flatclass option to the JVMOption1 custom properties variable in the **Properties** tab of the **Data Integration Service** page.
- Complete the following tasks on the machine where you installed the Informatica services:
 - Set the MAPR_HOME environment variable to the following path:
`<Server installation directory>/services/shared/hadoop/mapr_<version>`
 Verify the cluster details in the `mapr-clusters.conf` file that is located in the following directory:
`<Server installation directory>/services/shared/hadoop/mapr_<version>/conf`
 - Set the LD_LIBRARY_PATH variable to the following path:
`<Server installation directory>/services/shared/hadoop/mapr_<version>/lib/native/Linux-amd64-64`
- Complete the following tasks on the machine where you installed the Developer tool:
 - In the `run.bat` file, set the MAPR_HOME environment variable to the following path:
`<Client installation directory>\clients\DeveloperClient\hadoop\mapr_<version>`
Note: The `run.bat` file is located in the following directory:
`<Client installation directory>\clients\DeveloperClient`
 After you set the variable, double-click the `run.bat` file to launch the Developer tool.
 - Verify the cluster details in the `mapr-clusters.conf` file that is located in the following directory:
`<Client installation directory>\clients\DeveloperClient\hadoop\mapr_<version>\conf`
 - Add the -Dmapr.library.flatclass option to the `developerCore.ini` file.
 The `developerCore.ini` file is located in the following directory:

```
<Client installation directory>\clients\DeveloperClient\
```

- Add the following path to the `developerCore.ini` file:
`-Djava.library.path=hadoop\mapr_<version>\lib\native\Win32;bin;..\DT\bin`

Hortonworks Prerequisites

If you use the Hortonworks Hadoop distribution on Red Hat and Suse operating systems, you must configure the `LD_LIBRARY_PATH` variable.

Red Hat Prerequisites

If you use Red Hat operating systems, on the machine where you installed the Informatica services, set the `LD_LIBRARY_PATH` variable to the following path:

```
<Server installation directory>/services/shared/hadoop/hortonworks_<version>/lib/native
```

Suse Prerequisites

If you use Suse operating systems, on the machine where you installed the Informatica services, set the `LD_LIBRARY_PATH` variable to the following path:

```
<Server installation directory>/services/shared/hadoop/hortonworks_<version>/lib/suse_native
```

If the Hadoop cluster runs on Suse operating systems, you must also set the `LD_LIBRARY_PATH` environment variable in the `hadoopEnv.properties` file. The `LD_LIBRARY_PATH` environment variable in the `hadoopEnv.properties` file must include the native library path for a Hortonworks distribution. For example, set the `LD_LIBRARY_PATH` as follows:

```
infapdo.env.entry.ld_library_path=LD_LIBRARY_PATH=$HADOOP_NODE_INFA_HOME/services/shared/bin:  
$HADOOP_NODE_INFA_HOME/DataTransformation/bin:$HADOOP_NODE_HADOOP_DIST/lib/suse_native:  
$LD_LIBRARY_PATH
```

Note: By default, the `LD_LIBRARY_PATH` environment variable in the `hadoopEnv.properties` file is set for a Red Hat operating system.

CHAPTER 3

HDFS Connections

This chapter includes the following topics:

- [HDFS Connections Overview, 13](#)
- [HDFS Connection Properties, 13](#)
- [Creating an HDFS Connection, 14](#)

HDFS Connections Overview

Create an HDFS connection to read data from or write data to HDFS.

HDFS Connection Properties

Use a Hadoop File System (HDFS) connection to access data in the Hadoop cluster. The HDFS connection is a file system type connection. You can create and manage an HDFS connection in the Administrator tool, Analyst tool, or the Developer tool. HDFS connection properties are case sensitive unless otherwise noted.

Note: The order of the connection properties might vary depending on the tool where you view them.

The following table describes HDFS connection properties:

Property	Description
Name	Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /
ID	String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.
Description	The description of the connection. The description cannot exceed 765 characters.
Location	The domain where you want to create the connection. Not valid for the Analyst tool.
Type	The connection type. Default is Hadoop File System.

Property	Description
User Name	User name to access HDFS.
NameNode URI	<p>The URI to access HDFS.</p> <p>Use the following format to specify the NameNode URI in Cloudera and Hortonworks distributions: <code>hdfs://<namenode>:<port></code></p> <p>Where</p> <ul style="list-style-type: none"> - <namenode> is the host name or IP address of the NameNode. - <port> is the port that the NameNode listens for remote procedure calls (RPC). <p>Use one of the following formats to specify the NameNode URI in MapR distribution:</p> <ul style="list-style-type: none"> - <code>maprfs:///</code> - <code>maprfs:///mapr/my.cluster.com/</code> <p>Where <code>my.cluster.com</code> is the cluster name that you specify in the <code>mapr-clusters.conf</code> file.</p>

Creating an HDFS Connection

Create an HDFS connection before you import physical data objects.

1. Click **Window > Preferences**.
2. Select **Informatica > Connections**.
3. Expand the domain.
4. Select the connection type **File Systems > Hadoop File System**, and click **Add**.
5. Enter a connection name.
6. Optionally, enter a connection description.
7. Click **Next**.
8. Configure the connection properties.
9. Click **Test Connection** to verify the connection to HDFS.
10. Click **Finish**.

RELATED TOPICS:

- [“HDFS Connection Properties” on page 13](#)

CHAPTER 4

HDFS Data Objects

This chapter includes the following topics:

- [HDFS Data Objects Overview, 15](#)
- [Flat File Data Objects, 15](#)
- [Complex File Data Objects, 17](#)
- [Custom Formats, 21](#)

HDFS Data Objects Overview

After you configure an HDFS connection, create a physical data object to read data from or write data to HDFS.

Depending on the file format, you can configure the following types of physical data objects:

- Flat file data object. Create or import a flat file data object and configure an HDFS connection for the data object. Use the flat file data object to read or write unstructured data in fixed-width or delimited text files.
- Complex file data object. Import a complex file data object with an HDFS connection. Use the complex file data object to read or write semi-structured data in text, XML, JSON, sequence files, or binary files such as PDF, Microsoft Word, and Microsoft Excel.

Flat File Data Objects

You can read data from and write data to HDFS through a fixed-width or delimited flat file data object that does not contain binary data.

You can create or import a flat file data object. The data object properties that you specify in the Developer tool must match the properties of the source file.

After you create a flat file data object, you can edit the following file properties:

- HDFS connection properties
- Compression formats

You can compress and decompress files in formats such as Gzip and Bzip2. To read large volumes of data, you can connect a flat file source to read data from a directory of flat files.

You can use the flat file data objects as a source, target, or lookup transformation in mappings and mapplets. You can select the mapping environment and run the mappings in a native or Hive run-time environment. You can create and run profiles against flat file data objects.

When you configure a mapping that contains flat file data objects to run in the native environment, you can enable the mapping for partitioning. The Data Integration Service can use multiple partitions to read data from flat file sources with an HDFS connection. The Data Integration Service can also use multiple partitions to write data to flat file targets with an HDFS connection. When the Data Integration Service adds partitions, it increases the number of processing threads, which can increase mapping performance.

Compression and Decompression for Flat File Sources and Targets

File compression can increase data transfer rates and reduce space for data storage.

You can read and write compressed flat files, specify compression formats, and decompress files. You can compress and decompress files in compression formats such as Bzip2 and Lzo, or specify a custom compression format.

You can specify a file or a directory of files. When the Data Integration Service reads from a directory, it reads the files of the specified format only and ignores files of other formats.

Compressed files that you read in a Hive run-time environment cannot be split.

For information about how Hadoop processes compressed and uncompressed files, see the Hadoop documentation.

The following table describes the compression options:

Compression Options	Description
None	The file is not compressed.
Auto	The Data Integration Service detects the compression format of the file based on the file extension.
Gzip	The GNU zip compression format that uses the DEFLATE algorithm.
Bzip2	The Bzip2 compression format that uses the Burrows–Wheeler algorithm.
Lzo	The Lzo compression format that uses the Lempel-Ziv-Oberhumer algorithm. Note: The JAR files for LZO compression are not available with the default Hadoop installation. You must place the JAR files for the LZO compression format in the <code>lib</code> folder of the distribution directory and verify the distribution directory properties.
Custom	Custom compression format. If you select this option, you must specify the fully qualified class name implementing the Hadoop <code>CompressionCodec</code> interface in the Compression Codec field.

RELATED TOPICS:

- [“Custom Formats Configuration” on page 21](#)

Rules and Guidelines for Flat File Data Objects

Use the following rules and guidelines when you use flat file sources with an HDFS connection:

- You cannot use a command to generate or transform flat file data and send the output to the flat file source at run time.
- You cannot use an indirect source type.

Use the following rules and guidelines when you use flat file targets with an HDFS connection:

- You cannot append output data to target files and reject files. The Data Integration Service truncates the target files and reject files before writing the data.
- You cannot use the command output type.
- When the flat file target is in a partitioned mapping, you cannot write to a merge file that contains the target output for all partitions. The Data Integration Service concurrently writes the target output to a separate file for each partition.

Configuring a Flat File Data Object with an HDFS Connection

Configure a flat file data object with an HDFS connection to read data from or write data to HDFS.

If you create an empty flat file, the file properties must match that of the file in HDFS. If you import a flat file data object, the file must reside in your local file system.

1. Click the **Advanced** tab of the flat file data object.
2. Navigate to the runtime properties for the flat file source in the **Runtime: Read** properties or the flat file target in the **Runtime: Write** properties.
3. Configure the HDFS connection properties.
4. Optionally, you can configure the compression properties.

RELATED TOPICS:

- [“Flat File Data Object Read Properties” on page 22](#)
- [“Flat File Data Object Write Properties” on page 26](#)

Complex File Data Objects

A complex file data object is a representation of a file in the Hadoop file system. Create a complex file data object to read or write semi-structured data to HDFS.

You can read files from the local system or HDFS. Similarly, you can write files to the local system or HDFS. To read large volumes of data, you can connect a complex file source to read data from a directory of files that have the same format and properties. You can read and write compressed binary files.

The output of the complex file data object is a binary stream. When you read data from a complex file, you must use a Data Processor transformation to parse the file. Similarly, when you write data to a complex file, you must use a Data Processor transformation to convert the source data into a binary format. You can then use the binary stream to write data to the complex file.

When you create a complex file data object, a read and write operation is created. You can use the complex file data object read operation as a source in mappings and mapplets. You can use the complex file data object write operation as a target in mappings and mapplets. You can select the mapping environment and run the mappings in a native or Hive run-time environment.

Complex File Data Object Overview Properties

The Data Integration Service uses overview properties when it reads data from or writes data to a complex file.

Overview properties include general properties that apply to the complex file data object. They also include object properties that apply to the resources in the complex file data object. The Developer tool displays overview properties for complex files in the **Overview** view.

General Properties

The following table describes the general properties that you configure for complex files:

Property	Description
Name	The name of the complex file data object.
Description	The description of the complex file data object.
Access Method	The access method for the resource. <ul style="list-style-type: none">- Connection. Select Connection to specify an HDFS connection.- File. Select File to browse for a file on your local system.
Connection	The name of the HDFS connection.

Objects Properties

The following table describes the objects properties that you configure for complex files:

Property	Description
Name	The name of the resource.
Type	The native data type of the resource.
Description	The description of the resource.
Access Type	Indicates that you can perform read and write operations on the complex file data object. You cannot edit this property.

Compression and Decompression for Complex File Sources and Targets

You can read and write compressed complex files, specify compression formats, and decompress files. You can use compression formats such as Bzip2 and Lzo, or specify a custom compression format. The compressed files must be of the binary format.

You can compress sequence files at a record level or at a block level.

For information about how Hadoop processes compressed and uncompressed files, see the Hadoop documentation.

The following table describes the complex file compression formats for binary files:

Compression Options	Description
None	The file is not compressed.
Auto	The Data Integration Service detects the compression format of the file based on the file extension.
DEFLATE	The DEFLATE compression format that uses a combination of the LZ77 algorithm and Huffman coding.
Gzip	The GNU zip compression format that uses the DEFLATE algorithm.
Bzip2	The Bzip2 compression format that uses the Burrows–Wheeler algorithm.
Lzo	The Lzo compression format that uses the Lempel–Ziv–Oberhumer algorithm.
Snappy	The LZ77-type compression format with a fixed, byte-oriented encoding.
Custom	Custom compression format. If you select this option, you must specify the fully qualified class name implementing the <code>CompressionCodec</code> interface in the Custom Compression Codec field.

RELATED TOPICS:

- [“Custom Formats Configuration” on page 21](#)

Parameterization of Complex File Data Objects

You can parameterize the complex file connection and the complex file data object operation properties.

You can parameterize the following data object read operation properties for complex data objects:

- Connection in the run-time properties
- File Format, Input Format, Compression Format, and Custom Compression Codec in the advanced properties.

You can parameterize the following data object write operation properties for complex file data objects:

- Connection in the run-time properties.
- File Name, Output Format, Output Key Class, Output Value Class, Compression Format, Custom Compression Codec, and Sequence File Compression Type in the advanced properties.

The following attributes support full and partial parameterization for complex file data objects:

- File Path in the advanced properties of the data object read operation.
For example, to parameterize a part of the attribute value where the file path in the advanced property is `/user/adpqa/dynschema.txt`, create a parameter as `$str="/user/adpqa"`, and then edit the file path as `$str/dynschema.txt`. You can also parameterize the value of the entire file path.

- File Directory in the advanced properties of the data object write operation.
For example, to parameterize a part of the attribute value where the file directory in the advanced property is `/export/home/qamercury/source`, create a parameter as `$param="/export/home"`, and then edit the file directory as `$param/qamercury/source`. You can also parameterize the value of the entire directory.

Creating a Complex File Data Object

Create a complex file data object to read data from or write data to HDFS.

1. Select a project or folder in the **Object Explorer** view.
2. Click **File > New > Data Object**.
3. Select **Complex File Data Object** and click **Next**.
The **New Complex File Data Object** dialog box appears.
4. Click **Browse** next to the **Location** option and select the target project or folder.
5. In the **Access Method** list, select **Connection** or **File**.
 - Select **Connection** to access a file on HDFS. Click **Browse** next to the **Connection** option and select an HDFS connection. Click **Add** next to the **Selected Resource** option to add a resource to the data object. The **Add Resource** dialog box appears. Navigate or search for the resources to add to the data object and click **OK**.
 - Select **File** to access a file on your local system. Click **Browse** next to the **Resource Location** option and select the file that you want to add. Click **Fetch**. The selected file is added to the **Selected Resources** list.
6. Optionally, enter a name for the data object.
7. Click **Finish**.

The data object appears under the Physical Data Objects category in the project or folder in the **Object Explorer** view. A read and write operation is created for the data object. Depending on whether you want to use the complex file data object as a source or target, you can edit the read or write operation properties. You can also create multiple read and write operations for a complex file data object.

Complex File Data Object Output Parsing

When you use a complex file data object as a source, you must use a Data Processor transformation to parse the binary output of the complex file.

Configure the Data Processor transformation as follows:

- Set an input port to buffer input and binary data type. Specify the port size. The port size that you specify in the complex file properties and the Data Processor transformation must be the same.
- Set an output port to buffer output or set it for relational output. If you set the ports for relational output, specify the ports based on the number of relational groups of ports you want in the output. Specify the port size for the ports. You can use an XML schema reference that describes the XML hierarchy.
- Set a Streamer object as a startup component.

When you use a complex file data object as a target, you must use a Data Processor transformation to convert the source data into a binary format. Set the Data Processor transformation port to binary. You can then use the binary stream as an input to the complex file data object.

Custom Formats

Custom formats provide flexibility with the input, output, and compression formats that you can use with PowerExchange for HDFS.

Apart from the input, output, and compression formats that PowerExchange for HDFS supports, you can use custom formats to read, write, and compress files. You can use the custom formats that Hadoop supports.

You can specify the following custom formats:

- Custom input format for complex file data objects
- Custom output format for complex file data objects
- Custom compression format for flat file and complex file data objects

Custom Formats Configuration

Before you use custom formats, you must complete configuration tasks in the Informatica environment.

To use custom formats in the native environment, copy the JAR files that implement the custom formats to the following location:

```
<Informatica installation directory>/services/shared/hadoop/<hadoop distribution name>/  
infaLib
```

To use custom formats in the Hive environment, see the Hadoop documentation for information on the prerequisite tasks.

If the custom compression includes native libraries, depending on the run-time environment, add the path of the native libraries in the environment variable `$LD_LIBRARY_PATH` or in the `hadoopEnv.properties` file.

The `hadoopEnv.properties` file is located in the following directory:

```
<Client installation directory>\clients\DeveloperClient\hadoop\Hadoop distribution_<version>  
\infaConf
```

The following table describes how you must set the native libraries path based on the run-time environment:

Run-time Environment	Add the Path to
Native	<code>\$LD_LIBRARY_PATH</code>
Hive	<code>hadoopEnv.properties</code>

The Java library path in the `hadoopEnv.properties` file must include the native library path. For example, the following property in the `hadoopEnv.properties` file specifies a native library path for a Cloudera distribution:

```
infapdo.java.opts=-Djava.library.path=$HADOOP_NODE_INFA_HOME/services/shared/bin:  
$HADOOP_NODE_INFA_HOME/services/shared/hadoop/cloudera_cdh5u4/lib/native
```

Note: If you use Hortonworks or MapR distributions, change the native library path based on the distribution.

CHAPTER 5

HDFS Data Extraction

This chapter includes the following topics:

- [HDFS Data Extraction Overview, 22](#)
- [Flat File Data Object Read Properties, 22](#)
- [Complex Files Partitioning, 23](#)
- [Complex File Data Object Read Properties, 23](#)

HDFS Data Extraction Overview

You can use a flat file data object or a complex file data object to read data from HDFS.

Complete the following tasks to read data from HDFS by using PowerExchange for HDFS:

1. Create an HDFS connection.
2. Create a flat file data object or a complex file data object. Specify the data object properties such as the file location, compression format, and input format.
3. Create a mapping and use the flat file data object or the complex file data object read operation as a source.
4. Configure a Data Processor transformation to parse the complex file.
5. Configure the validation and run-time environment type.
6. Run the mapping to read data from HDFS.

Flat File Data Object Read Properties

The Data Integration Service uses read properties when it reads data from a flat file. You can edit the format and runtime read properties on the **Advanced** tab.

The following table describes the HDFS connection and compression run-time properties that you configure for flat file sources:

Property	Description
Connection Type	The type of connection. Select from the following options: <ul style="list-style-type: none">- None. The source file does not require a connection.- Hadoop File System. The source file resides in HDFS. Default is None.
Connection Name	The name of the connection. Select an HDFS connection or assign a mapping parameter that defines the connection details.
Compression Format	Optional. Specifies the compression format. Select from the following options: <ul style="list-style-type: none">- None- Auto- Gzip- Bzip2- Lzo- Custom
Compression Codec	Required for custom compression. Specify the fully qualified class name implementing the <code>Hadoop CompressionCodec</code> interface.

Complex Files Partitioning

When you run a mapping in a Hive environment to read data from sequence files and custom input format files that are splittable, the Data Integration Service uses multiple partitions to read data from the source. The Data Integration Service creates multiple Map jobs to read data in parallel, thereby resulting in high performance.

To read text files in parallel, specify the following input format in the complex file read properties:

```
com.informatica.adapter.hdfs.hadoop.io.InfaTextInputFormat
```

You can also specify the following input format to read text files in batches:

```
com.informatica.adapter.hdfs.hadoop.io.InfaBatchTextInputFormat
```

Typically, when you read complex files, the Data Processor transformation has a Streamer component and a Parser component. By default, the Data Integration Service calls the Data Transformation Engine for every record. You can modify this behavior by using the count property in the Streamer component. Set the count property to define the number of records that the Data Integration Service must treat as a batch. When you set the count property, the Data Integration Service calls the Data Transformation Engine for each batch of records instead of calling the Data Transformation Engine for every record. Since the Data Integration Service processes the text files in batches, the performance increases.

Complex File Data Object Read Properties

The Data Integration Service uses read properties when it reads data from a complex file. Select the Output transformation to edit the general, ports, sources, and run-time properties.

General Properties

The Developer tool displays general properties for complex file sources in the **Read** view.

The following table describes the general properties that you configure for complex file sources:

Property	Description
Name	The name of the complex file. This property is read-only. You can edit the name in the Overview view. When you use the complex file as a source in a mapping, you can edit the name in the mapping.
Description	The description of the complex file.

Ports Properties

Ports properties for a physical data object include port names and port attributes such as data type and precision.

Note: The port size specified in the source transformation and Output transformation must be the same.

The following table describes the ports properties that you configure for complex file sources:

Property	Description
Name	The name of the resource.
Type	The native data type of the resource.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Description	The description of the resource.

Sources Properties

The Developer tool displays the sources properties for complex file sources in the Output transformation in the **Read** view.

The sources properties list the resources of the complex file data object. You can add or remove resources in the data object.

Advanced Properties

The Developer tool displays the advanced properties for complex file sources in the Output transformation in the **Read** view.

The following table describes the advanced properties that you configure for complex file sources:

Property	Description
File path	<p>The location of the file or directory. If the path is a directory, all the files in the directory must have the same file format.</p> <p>If the file or directory is in HDFS, enter the path without the node URI. For example, <code>/user/lib/testdir</code> specifies the location of a directory in HDFS. The path must not contain more than 512 characters.</p> <p>If the file or directory is in the local system, enter the fully qualified path. For example, <code>/user/testdir</code> specifies the location of a directory in the local system.</p> <p>Note: The Data Integration Service ignores any subdirectories and their contents.</p>
File Format	<p>The file format. Select one of the following file formats:</p> <ul style="list-style-type: none">- Binary. Select Binary for text, XML, JSON, or binary files such as PDF, Microsoft Word, and Microsoft Excel.- Sequence File Format. Select Sequence File Format for source files of a Hadoop-specific binary format that contain key and value pairs.- Input File Format. Select Input File Format to specify a custom input format. You must specify the class name implementing the <code>InputFormat</code> interface in the Input Format field. <p>Default is Binary.</p>
Input Format	<p>The class name for files of the input file format. If you select Input File Format in the File Format field, you must specify the fully qualified class name implementing the <code>InputFormat</code> interface.</p> <p>To read files that use the Avro format, use the following input format:</p> <pre>com.informatica.avro.AvroToXML</pre> <p>To read files that use the Parquet format, use the following input format:</p> <pre>com.informatica.parquet.ParquetToXML</pre> <p>You can use any class derived from <code>org.apache.hadoop.mapreduce.InputFormat</code>.</p>
Compression Format	<p>Optional. The compression format for binary files. Select one of the following options:</p> <ul style="list-style-type: none">- None- Auto- DEFLATE- gzip- bzip2- Lzo- Snappy- Custom
Custom Compression Codec	<p>Required for custom compression. Specify the fully qualified class name implementing the <code>CompressionCodec</code> interface.</p>

CHAPTER 6

HDFS Data Load

This chapter includes the following topics:

- [HDFS Data Load Overview, 26](#)
- [Flat File Data Object Write Properties, 26](#)
- [Complex File Streaming, 27](#)
- [Complex Files Output Collection Mode, 28](#)
- [Complex File Data Object Write Properties, 29](#)

HDFS Data Load Overview

You can use a flat file data object or a complex file data object to write data to HDFS.

Complete the following tasks to write data to HDFS by using PowerExchange for HDFS:

1. Create an HDFS connection.
2. Create a flat file data object or a complex file data object. Specify the data object properties such as the file location and compression format.
3. Create a mapping and use the flat file data object or the complex file data object write operation as a target.
4. Configure the validation and run-time environment type.
5. Run the mapping to write data to HDFS.

Flat File Data Object Write Properties

The Data Integration Service uses write properties when it writes data to a flat file. You can edit the format and runtime write properties on the **Advanced** tab.

The following table describes the HDFS connection and compression properties that you configure for flat file targets:

Property	Description
Connection Type	The type of connection. Select from the following options: <ul style="list-style-type: none"> - None. The target file does not require a connection. The target file location is specified by the output file directory. - Hadoop File System. The target file is in HDFS. Default is None.
Connection Name	The name of the connection. Select an HDFS connection or assign a mapping parameter that defines the connection details.
Compression Format	Optional. Specifies the compression format. Select from the following options: <ul style="list-style-type: none"> - None - Gzip - Bzip2 - Lzo - Custom
Compression Codec	Required for custom compression. Specify the fully qualified class name implementing the Hadoop <code>CompressionCodec</code> interface.

Complex File Streaming

To write data to a complex file, include a Data Processor transformation in the mapping to convert the source data into a binary format. You can use the binary stream to write data to the complex file.

The Data Processor transformation continually streams and sends input to the complex file target. It sends end of file information after it fully streams a file. It sends end of streaming information when it streams the entire input fully.

When the Data Processor transformation sends portions of the input to the complex file target, PowerExchange for HDFS appends unique identifier information to the file name. The Data Integration Service uses the unique identifiers to recognize that the streaming is in progress and not complete. Therefore, the file name that you specify in the complex file write properties is not the same as the output file in HDFS. The output file name in HDFS contains the unique identifier information as well.

The unique identifier format depends on the run-time environment that you select for the mapping and on whether the file is not compressed or not.

The following table describes the unique identifier format based on the run-time environment and on whether the file is not compressed or not:

Run-time Environment Type	File Type	Unique Identifier Format
Native	Uncompressed File	<filename>_<unique identifier>_<seq>.<ext>
Native	Compressed File	<filename>_<unique identifier>_<seq>.<compression format extension>

Run-time Environment Type	File Type	Unique Identifier Format
Hive	Uncompressed File	<filename>-<reducer ID>_<unique identifier>_<seq>.<ext>
Hive	Compressed File	<filename>_<reducer ID>_<unique identifier>_<seq>.<compression format extension>

If you do not include the compression format extension as part of the file name in the complex file write properties, PowerExchange for HDFS appends extensions based on the compression format.

The following table describes the extensions that PowerExchange for HDFS appends based on the compression format that you use:

Compression Format	File Name Extension that PowerExchange for HDFS Appends
DEFLATE	.deflate
Gzip	.gz
Bzip2	.bz2
Lzo	.lzo
Snappy	.snz

Complex Files Output Collection Mode

When you write data to complex files, you can choose to collect the input rows and write the output to a single file, or create an output row for each input row.

You can specify the output collection mode in the Data Processor transformation or in the `hive-site.xml` file based on the complex file type.

To specify the output collection mode in the Data Processor transformation, open the Data Processor transformation and click the **Settings** view. In the **Binary output collection mode** section, specify the output collection mode.

The following table describes the options that you can select for the output collection mode:

Property Name	Property Description
Collect input rows to a single output	Select this option if you want to collect all input rows and write the output to a single file.
Split output when size exceeds	When you write the output to a single file, you can choose to split the output file when it exceeds a particular size. Enter the size in MB exceeding which the file must be split. Default is 100 MB.
Output row for each input row (do not collect)	Select this option if you want to write an output row for each input row.

You can also use the `hive-site.xml` file to configure the number of output files that you want. Configure the number of reducers based on the number of output files that you want. For more information about configuring the `hive-site.xml` file, see the Hadoop documentation.

Output Collection Mode for Binary Files

When you write to binary files in a native or Hive environment, you can specify the output collection mode in the Data Processor transformation. When you write to binary files in a Hive environment, you can also use the `hive-site.xml` file to configure the number of output files that you want.

Output Collection Mode for Sequence Files and Custom Output Format Files

When you write to sequence files or custom output format files in a native environment, PowerExchange for HDFS writes all the key-value pairs into one output file. The number of key-value pairs that PowerExchange for HDFS writes depends on the output collection mode that you specified in the Data Processor transformation.

When you write to sequence files or custom output format files in a Hive environment, you can use the `hive-site.xml` file to configure the number of output files that you want. The number of key-value pairs that PowerExchange for HDFS writes depends on the output collection mode that you specified in the Data Processor transformation.

Complex File Data Object Write Properties

The Data Integration Service uses write properties when it writes data to a complex file. Select the Input transformation to edit the general, ports, sources, and advanced properties.

General Properties

The Developer tool displays general properties for complex file targets in the **Write** view.

The following table describes the general properties that you configure for complex file targets:

Property	Description
Name	The name of the complex file. This property is read-only. You can edit the name in the Overview view. When you use the complex file as a target in a mapping, you can edit the name in the mapping.
Description	The description of the complex file.

Port Properties

Port properties for a physical data object include port names and port attributes such as data type and precision.

Note: The port size specified in the target transformation and Input transformation must be the same.

The following table describes the ports properties that you configure for complex file targets:

Property	Description
Name	The name of the resource.
Type	The native data type of the resource.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Description	The description of the resource.

Sources Properties

The Developer tool displays the sources properties for complex file targets in the Input transformation in the **Write** view.

The sources properties list the resources of the complex file data object. You can add or remove resources in the data object.

Advanced Properties

The Developer tool displays the advanced properties for complex file targets in the Input transformation in the **Write** view.

The following table describes the advanced properties that you configure for complex file targets:

Property	Description
File Directory	<p>The directory location of the complex file target.</p> <p>If the directory is in HDFS, enter the path without the node URI. For example, <code>/user/lib/testdir</code> specifies the location of a directory in HDFS. The path must not contain more than 512 characters.</p> <p>If the directory is in the local system, enter the fully qualified path. For example, <code>/user/testdir</code> specifies the location of a directory in the local system.</p> <p>Note: The Data Integration Service ignores any subdirectories and their contents.</p>
File Name	<p>The name of the output file. PowerExchange for HDFS appends the file name with a unique identifier before it writes the file to HDFS.</p>
File Format	<p>The file format. Select one of the following file formats:</p> <ul style="list-style-type: none">- Binary. Select Binary for text, XML, JSON, or binary files such as PDF, Microsoft Word, and Microsoft Excel.- Sequence File Format. Select Sequence File Format for target files of a Hadoop-specific binary format that contain key and value pairs.- Output Format. Select Output Format to specify a custom output format. You must specify the class name implementing the <code>OutputFormat</code> interface in the Output Format field. <p>Default is Binary.</p>
Output Format	<p>The class name for files of the output format. If you select Output Format in the File Format field, you must specify the fully qualified class name implementing the <code>OutputFormat</code> interface.</p>
Output Key Class	<p>The class name for the output key. If you select Output Format in the File Format field, you must specify the fully qualified class name for the output key.</p> <p>You can specify one of the following output key classes:</p> <ul style="list-style-type: none">- BytesWritable- Text- LongWritable- IntWritable <p>Note: PowerExchange for HDFS generates the key in ascending order.</p>
Output Value Class	<p>The class name for the output value. If you select Output Format in the File Format field, you must specify the fully qualified class name for the output value.</p> <p>You can use any custom writable class that Hadoop supports. Determine the output value class based on the type of data that you want to write.</p> <p>Note: When you use custom output formats, the value part of the data that is streamed to the complex file data object write operation must be in a serialized form.</p>
Compression Format	<p>Optional. The compression format for binary files. Select one of the following options:</p> <ul style="list-style-type: none">- None- Auto- DEFLATE- gzip- bzip2- LZ0- Snappy- Custom

Property	Description
Custom Compression Codec	Required for custom compression. Specify the fully qualified class name implementing the <code>CompressionCodec</code> interface.
Sequence File Compression Type	Optional. The compression format for sequence files. Select one of the following options: <ul style="list-style-type: none"> - None - Record - Block

CHAPTER 7

HDFS Mappings

This chapter includes the following topics:

- [HDFS Mappings Overview, 33](#)
- [Hive Validation and Run-time Environments, 33](#)
- [HDFS Data Extraction Mapping Example, 34](#)
- [HDFS Data Load Mapping Example, 35](#)

HDFS Mappings Overview

After you create a flat file or a complex file data object operation, you can create an HDFS mapping.

You can define the following objects in an HDFS mapping:

- A flat file data object or a complex file data object read operation as the input to read data from HDFS
- Transformations
- A flat file data object or a complex file data object write operation as the output to write data to HDFS

If you use a complex file data object as a source, you must use a Data Processor transformation to parse the file. Similarly, when you use a complex file data object as a target, you must use a Data Processor transformation to convert the source data into a binary format. You can then use the binary stream to write data to the complex file.

Validate and run the mapping. You can deploy the mapping and run it or add the mapping to a Mapping task in a workflow and run the workflow. You can also run the mapping in a Hive run-time environment.

Hive Validation and Run-time Environments

You can use flat file and complex file data objects in a Hive run-time environment.

You can configure the mappings to run in native or Hive run-time environments. When you run a mapping in the native environment, the Data Integration Service processes the mapping. When you run a mapping in a Hive environment, the Data Integration Service can push mappings to a Hadoop cluster.

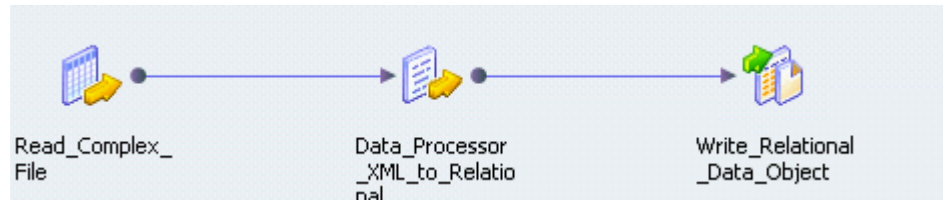
For more information about the Hive run-time and validation environment, see *Big Data Management User Guide*.

HDFS Data Extraction Mapping Example

Your organization needs to analyze purchase order details such as customer ID, item codes, and item quantity. The purchase order details are stored in a semi-structured compressed XML file in HDFS. The hierarchical data includes a purchase order parent hierarchy level and a customer contact details child hierarchy level. Create a mapping that reads all the purchase records from the file in HDFS. The mapping must convert the hierarchical data to relational data and write it to a relational target.

You can use the extracted data for business analytics.

The following figure shows the example mapping:



You can use the following objects in the HDFS mapping:

HDFS Input

The input, Read_Complex_File, is a compressed XML file stored in HDFS.

Data Processor Transformation

The Data Processor transformation, Data_Processor_XML_to_Relational, parses the XML file and provides a relational output.

Relational Output

The output, Write_Relational_Data_Object, is a table in an Oracle database.

When you run the mapping, the Data Integration Service reads the file in a binary stream and passes it to the Data Processor transformation. The Data Processor transformation parses the specified file and provides a relational output. The output is written to the relational target.

You can configure the mapping to run in a native or Hadoop run-time environment.

Complete the following tasks to configure the mapping:

1. Create an HDFS connection to read files from the Hadoop cluster.
2. Create a complex file data object read operation. Specify the following parameters:
 - The file as the resource in the data object.
 - The file compression format.
 - The HDFS file location.
3. Optionally, you can specify the input format that the Mapper uses to read the file.
4. Drag and drop the complex file data object read operation into a mapping.
5. Create a Data Processor transformation. Configure the following properties in the Data Processor transformation:
 - An input port set to buffer input and binary data type.
 - Relational output ports depending on the number of columns you want in the relational output. Specify the port size for the ports. Use an XML schema reference that describes the XML hierarchy. Specify the normalized output that you want. For example, you can specify

PurchaseOrderNumber_Key as a generated key that relates the Purchase Orders output group to a Customer Details group.

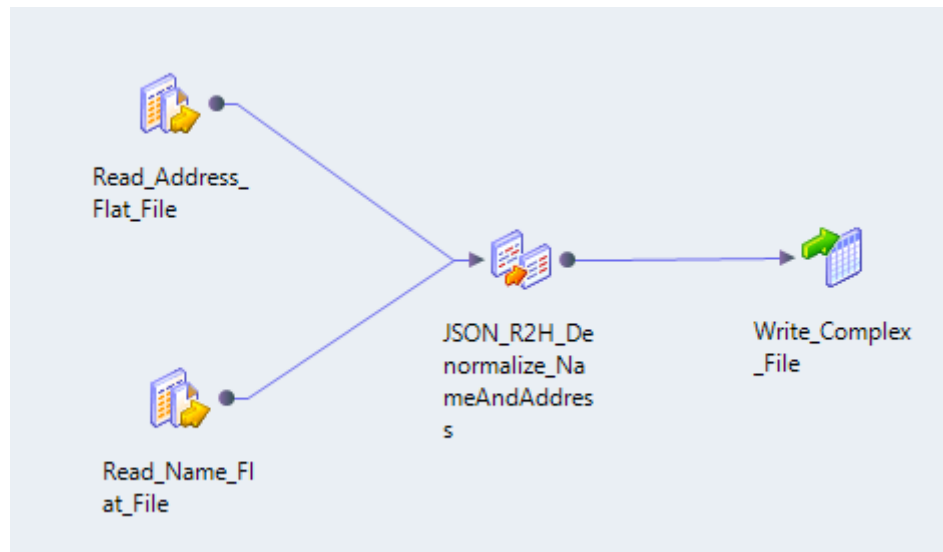
- Create a Streamer object and specify Streamer as a startup component.
6. Create a relational connection to an Oracle database.
 7. Import a relational data object.
 8. Create a write transformation for the relational data object and add it to the mapping.

HDFS Data Load Mapping Example

Your organization needs to denormalize employee ID, name, and address details. The employee ID, name, and address details are stored in flat files in HDFS. Create a mapping that reads all the employee ID, name, and address details from the flat files in HDFS. The mapping must convert the denormalized data to hierarchical data and write it to a complex file target in HDFS.

You can use the target data for business analytics.

The following figure shows the example mapping:



You can use the following objects in the HDFS mapping:

HDFS Inputs

The inputs, Read_Address_Flat_File and Read_Name_Flat_File, are flat files stored in HDFS.

Data Processor Transformation

The Data Processor transformation, JSON_R2H_Denormalize_NameAndAddresses, reads the flat files, denormalizes the data, and provides a binary, hierarchical output.

HDFS Output

The output, Write_Complex_File, is a complex file stored in HDFS.

When you run the mapping, the Data Integration Service reads the input flat files and passes the employee ID, name, and address data to the Data Processor transformation. The Data Processor transformation denormalizes the employee ID, name, and address data, and provides a hierarchical output in a binary stream. The binary and hierarchical output is written to the HDFS complex file target.

You can configure the mapping to run in a native or Hive run-time environment.

Complete the following tasks to configure the mapping:

1. Create an HDFS connection to read flat files from the Hadoop cluster.
2. Specify the read properties for the flat files.
3. Drag and drop the flat files into a mapping.
4. Create a Data Processor transformation. Set the Data Processor transformation port to binary.
5. Create an HDFS connection to write data to the complex file target.
6. Create a complex file data object write operation. Specify the following parameters:
 - The file as the resource in the data object.
 - The HDFS file location.
7. Drag and drop the complex file data object write operation into the mapping.

APPENDIX A

Data Type Reference

This appendix includes the following topics:

- [Data Type Reference Overview, 37](#)
- [Flat File and Transformation Data Types, 37](#)
- [Complex File and Transformation Data Types, 38](#)

Data Type Reference Overview

Informatica Developer uses the following data types for HDFS data objects:

- Flat file data types. Flat file data types appear in the physical data object column properties.
- Complex file data types. Complex file data types appear in the physical data object column properties.
- Transformation data types. Set of data types that appear in the transformations. They are internal data types based on ANSI SQL-92 generic data types, which the Data Integration Service uses to move data across platforms. Transformation data types appear in all transformations in a mapping.

When the Data Integration Service reads source data, it converts the native data types to the comparable transformation data types before transforming the data. When the Data Integration Service writes to a target, it converts the transformation data types to the comparable native data types.

Flat File and Transformation Data Types

The following table lists the flat file data types that the Data Integration Service supports and the corresponding transformation data types:

Flat File Data type	Transformation Data type	Range
Bigint	Bigint	Precision of 19 digits, scale of 0
Datetime	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Double	Double	Precision of 15 digits

Flat File Data type	Transformation Data type	Range
Int	Integer	-2,147,483,648 to 2,147,483,647
Nstring	String	1 to 104,857,600 characters
Number	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
String	String	1 to 104,857,600 characters

When the Data Integration Service reads non-numeric data in a numeric column from a flat file, it drops the row and writes a message in the log. Also, when the Data Integration Service reads non-datetime data in a datetime column from a flat file, it drops the row and writes a message in the log.

Complex File and Transformation Data Types

Complex file data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table lists the complex file data types that the Data Integration Service supports and the corresponding transformation data types:

Complex File Data Type	Transformation Data Type	Range and Description
Binary	Binary	1 to 104,857,600 bytes. You can read and write data of Binary data type in a Hive environment. You can use the user-defined functions to transform the binary data.

INDEX

C

- complex file data objects
 - creating [20](#)
 - general properties [18](#)
 - objects properties [18](#)
 - overview [18](#)
- complex file output
 - parsing [20](#)
- complex file read properties
 - advanced properties [25](#)
 - general properties [24](#)
 - overview [23](#)
 - ports properties [24](#)
 - sources properties [24](#)
- complex file write properties
 - advanced properties [31](#)
 - general properties [30](#)
 - overview [29](#)
 - ports properties [30](#)
 - sources properties [30](#)
- complex files
 - compression [18](#)
 - decompression [18](#)
 - input formats for text files [23](#)
 - output collection mode [28](#)
 - partitioning [23](#)
 - streaming [27](#)
- custom formats
 - configuration [21](#)
 - overview [21](#)

D

- Data Processor transformation
 - configuration [20](#)
- data type reference
 - complex files [38](#)
 - flat files [37](#)
- data Type reference
 - overview [37](#)

F

- flat file data objects
 - compression [16](#)

- flat file data objects (*continued*)
 - configuring an HDFS connection [17](#)
 - decompression [16](#)
 - partitioning [15](#)
 - read properties [22](#)
 - rules and guidelines for using [17](#)
 - write properties [26](#)

H

- HDFS connections
 - creating [14](#)
 - overview [13](#)
 - properties [13](#)
- HDFS data objects
 - complex file data objects [17](#)
 - flat file data objects [15](#)
 - overview [15](#)
- HDFS mappings
 - data extraction example [34](#)
 - data load example [35](#)
 - overview [33](#)

M

- mapping run-time environment
 - Hive [33](#)
- mapping validation environment
 - Hive [33](#)

P

- PowerExchange for HDFS
 - data extraction [22](#)
 - data load [26](#)
 - overview [9](#)
- PowerExchange for HDFS configuration
 - Hortonworks prerequisites [12](#)
 - MapR prerequisites [11](#)
 - overview [10](#)
 - prerequisites [10](#)