Informatica® Cloud Data Integration

NetSuite Connector
Informatica Cloud Data Integration NetSuite Connector
February 2020

© Copyright Informatica LLC 2007, 2020

This software and documentation are provided only under a separate license agreement containing restrictions on use and disclosure. 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.

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation is subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License.

Informatica, the informatica logo, Informatica Cloud, and PowerCenter are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at https://www.informatica.com/trademarks.html. 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. Required third party notices are included with the product.


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.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, report them to us at infa_documentation@informatica.com.

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided. INFORMATICA PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Publication Date: 2020-10-14
# Table of Contents

## Chapter 1: Introduction to NetSuite Connector ........................................... 5
- NetSuite Connector overview. ................................................................. 5
- Changed data capture. ............................................................................. 6
- Data Integration Hosted Agent. ................................................................. 6
- NetSuite Connector administration. ......................................................... 7
  - Mapping NetSuite record field names. ................................................... 7
  - Configuring the NetSuiteCustomFields.ini file. ...................................... 9
  - Configuring the NetSuiteSavedSearchFields.ini file. .............................. 10
  - Increasing heap size for concurrency in synchronization tasks. ............... 13
  - Handling parent and child or sibling objects with similar column name. .... 14

## Chapter 2: NetSuite connections ............................................................... 15
- NetSuite connections overview. ............................................................... 15
- NetSuite token-based authentication. ......................................................... 15
  - Generating a NetSuite token ID and token secret. .................................. 16
- NetSuite connection properties. ............................................................... 16
- NetSuite WSDL URL versions. ................................................................. 20
- Rules and guidelines for NetSuite connections. ........................................ 20
  - Configuring NetSuite account-specific service URL. ............................. 20
- Troubleshooting a NetSuite connection. ................................................. 21

## Chapter 3: Synchronization tasks with NetSuite ........................................ 22
- NetSuite sources in synchronization tasks. .............................................. 22
- NetSuite targets in synchronization tasks. .............................................. 23
- NetSuite lookups in synchronization tasks. .............................................. 23
- Synchronization task schedule and advanced options for NetSuite. .......... 24
- Rules and guidelines for NetSuite objects in a synchronization task. .......... 25

## Chapter 4: Mappings and mapping tasks with NetSuite ............................. 27
- NetSuite objects in mappings. ............................................................... 27
  - NetSuite sources in mappings. ........................................................... 27
  - NetSuite targets in mappings. ........................................................... 29
  - NetSuite lookups in mappings. ........................................................... 31
- NetSuite objects in template-based mapping tasks. .................................. 31
  - NetSuite sources in mapping tasks. ..................................................... 31
  - NetSuite targets in mapping tasks. ..................................................... 31
- Rules and guidelines for NetSuite objects. ............................................ 32

## Appendix A: NetSuite data type reference ............................................. 34
- NetSuite data type reference overview. ................................................ 34
NetSuite and transformation data types. ........................................ 34
List/Record data type. ............................................................ 36

Index. .................................................................................... 37
CHAPTER 1

Introduction to NetSuite Connector

This chapter includes the following topics:

- NetSuite Connector overview, 5
- Changed data capture, 6
- Data Integration Hosted Agent, 6
- NetSuite Connector administration, 7

NetSuite Connector overview

NetSuite Connector enables you to securely read data from or write data to NetSuite. NetSuite sources and targets represent records in NetSuite. NetSuite records are tables that correspond to the tabs and other user interface elements on the NetSuite web site. For example, the Account record contains information for the fields in the NetSuite Accounts page.

You can use NetSuite objects as sources, lookups, and targets in synchronization tasks, PowerCenter tasks, mappings, and mapping tasks. When you use NetSuite objects in synchronization tasks, mappings, and mapping tasks, you must configure properties specific to NetSuite. When you use NetSuite objects in PowerCenter tasks, you do not configure properties specific to NetSuite.

You can work with the following types of NetSuite objects in Data Integration:

**Standard objects**

The standard object type for NetSuite. Standard objects, such as Accounts and Customer, are packaged within NetSuite.

**Custom objects**

A standard object that is customized to hold additional data. NetSuite system administrators and users with required permissions can define custom fields for standard and custom records.

**Saved searches**

A reusable search definition that can include one or more advanced search criteria and search results. You can use standard join to combine two or more standard objects in saved searches. You can use custom join to combine standard objects with custom objects in saved searches.
**Advanced search**

An ad hoc search definition that can include one or more advanced search criteria and search results. You can read data from standard fields and custom fields of NetSuite advanced search that contains one or more siblings.

You can apply filter on advanced search.

**Note:** You can use NetSuite Connector with version 2019_2_0 of the NetSuite WSDL URL to read data from NetSuite advanced search.

You can use NetSuite standard objects, custom objects, and saved searches as a source. You can use NetSuite standard objects and custom objects as a target.

---

**Changed data capture**

You can capture changes in a NetSuite source and extract the changed data. To capture changed data in NetSuite, specify a start date and end date in the advanced properties for a NetSuite source used in a synchronization task, mapping, or mapping task.

When you capture changed data, use the following guidelines:

- Specify the start date and end date in the valid format.
- Specify a time period in the past.
- Specify a start date before the end date.

When capturing changed data, the agent performs the following tasks:

- Reads data that was created within the specified time period and marks them for insert.
- Reads data that was updated within the specified time period and marks them for update.
- Reads data that was deleted within the specified time period and marks them for delete.

---

**Data Integration Hosted Agent**

You can use the Data Integration Hosted Agent (Hosted Agent) as a runtime environment for a NetSuite connection if you have the Cloud Runtime license.

Data Integration Secure Agents are installed locally. As an alternative to installing a Secure Agent, you can use a Hosted Agent. Hosted Agents are hosted at a Data Integration hosting facility. The Data Integration hosting facility manages the Hosted Agent runtime environment and the agents that run in it. You cannot add, delete, or configure a Hosted Agent runtime environment. Because you do not install a Hosted Agent, you do not have access to files normally stored in the Secure Agent directory, such as configuration, success, and reject files.
NetSuite Connector administration

The administrator can perform the following tasks to ensure that NetSuite fields are available in Data Integration and to optimize performance:

- If field names in the NetSuite record and the related NetSuite search record do not match, the fields cannot be used in filters. This must be configured before NetSuite objects can be used as sources in tasks.
  You can associate NetSuite record field names with related NetSuite search record field names when you set up a NetSuite connection in Data Integration. Alternatively, you can configure the RecordToFieldsMap.ini file if the connection uses a Secure Agent group for the runtime environment.
- Some custom fields in NetSuite custom objects might not appear in Data Integration, particularly transaction body fields and transaction column fields.
  To ensure that all custom fields are available in Data Integration, you can configure custom fields when you set up a NetSuite connection in Data Integration. Alternatively, you can configure the NetSuiteCustomFields.ini file if the connection uses a Secure Agent group for the runtime environment.
- Saved search fields in NetSuite do not appear in Data Integration if they have a null value.
  You can specify saved search fields when you configure a NetSuite connection so that the saved search fields appear in Data Integration even when they have a null value. Alternatively, you can add the saved search fields to the NetSuiteSavedSearchFields.ini file if the connection uses a Secure Agent group for the runtime environment.
- To optimize performance for concurrent threads in synchronization tasks, you might need to adjust heap size.

Mapping NetSuite record field names

Map NetSuite record field names with related NetSuite search record field names so that users can use the fields in filters.

Users define filters for fields in the following locations:

- **Data Filters** page of the Synchronization Task wizard
- Query options in a Source transformation in the Mapping Designer
- Query options in the **Sources** page of the Mapping Task wizard when the source object is parameterized

NetSuite SearchBasic search records contain the field names used for filtering. When a field name in a NetSuite record matches the related field name in the corresponding SearchBasic search record, users can define a filter for the field in tasks. When a field name in a record does not match the related search record field name, users cannot define a filter for the field in tasks.

You can check the NetSuite schema to see if a field name in a NetSuite record matches the related field name in the SearchBasic search record. NetSuite records are represented as schema types in the NetSuite schema. To view NetSuite field names in the NetSuite schema, you can open the appropriate schema definition file (XSD). You can find a list of XSDs in the following location:

  [https://webservice.netsuite.com/wsd/2019_2.0/netsuite.wsdl](https://webservice.netsuite.com/wsd/2019_2.0/netsuite.wsdl)

For example, a user wants to define a filter for account name. You review the Account record in NetSuite’s browser, which is defined in NetSuite’s **accounting.xsd** file.
The schema contains the field, acctName, as shown in the following sample:

```xml
<complexType name="Account">
  <complexContent>
    <extension base="platformCore:Record">
      <sequence>
        <element name="acctType" minOccurs="0" type="listAcctTyp:AccountType"/>
        <element name="unitsType" minOccurs="0" type="platformCore:RecordRef"/>
        <element name="unit" minOccurs="0" type="platformCore:RecordRef"/>
        <element name="acctNumber" minOccurs="0" type="xsd:string"/>
        <element name="includeChildren" minOccurs="0" type="xsd:boolean"/>
        <element name="currency" minOccurs="0" type="platformCore:RecordRef"/>
        <element name="exchangeRate" minOccurs="0" type="xsd:string"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

You also see that the corresponding SearchBasic field name in the NetSuite's common.xsd file is called name, as shown in the following sample:

```xml
<complexType name="AccountSearchRowBasic">
  <complexContent>
    <extension base="platformCore:SearchRowBasic">
      <sequence>
        <element name="balance" minOccurs="0" type="platformCore:SearchColumnsDoubleField" maxOccurs="unbounded"/>
        <element name="cashFlowType" minOccurs="0" type="platformCore:SearchColumnsSelectField" maxOccurs="unbounded"/>
        <element name="category199Misc" minOccurs="0" type="platformCore:SearchColumnsSelectField" maxOccurs="unbounded"/>
        <element name="description" minOccurs="0" type="platformCore:SearchColumnsStringField" maxOccurs="unbounded"/>
        <element name="externalId" minOccurs="0" type="platformCore:SearchColumnsSelectField" maxOccurs="unbounded"/>
        <element name="generalId" minOccurs="0" type="platformCore:SearchColumnsSelectField" maxOccurs="unbounded"/>
        <element name="internalId" minOccurs="0" type="platformCore:SearchColumnsSelectField" maxOccurs="unbounded"/>
        <element name="name" minOccurs="0" type="platformCore:SearchColumnsStringField" maxOccurs="unbounded"/>
        <element name="number" minOccurs="0" type="platformCore:SearchColumnsStringField" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Because the field name is acctName in the Account record and the field name is name in the corresponding SearchBasic record, you map the two field names. The following example shows how you map acctName to name in the connection properties:

```
[Account]
acctName=name
```

Use the same syntax if you resolve the issue in the RecordToFieldsMap.ini file.

**Creating a configuration file to map NetSuite fields**

To ensure that users can define filters for NetSuite fields, map record field names with related NetSuite search record field names. You can map the fields in the Record Filter Fields connection property when you configure a NetSuite connection. Alternatively, if the NetSuite connection uses a Secure Agent group as the runtime environment, you can create a configuration file to map the fields.

If you choose to create a configuration file to map the fields, perform the following steps:

1. Create a text file named RecordToFieldsMap.ini.
2. Use the following guidelines to configure the RecordToFieldsMap.ini file:
   - Create a separate section for each NetSuite record.
   - In each section, list the record field names and related SearchBasic field names, as follows:

   ```ini
   [Account]
   <record field name>=<SearchBasic field name>
   <record field name2>=<SearchBasic field name2>
   
   [Record 2]
   <record field name>=<SearchBasic field name>
   <record field name2>=<SearchBasic field name2>
   <record field name3>=<SearchBasic field name3>
   ```
For example:

[Account]
acctName=name
addr1=address1

- To read transactional data from NetSuite when memorized transaction is enabled in the NetSuite account, add the record field names and related SearchBasic field name in the following format:

  [record 1]
  <record field name>=<SearchBasic field name>

For example:

If you want to enable memorized transaction for JournalEntry object, add the following value in the RecordToFieldsMap.ini file:

[JournalEntry]
reversalEntry=memorized

3. Copy the RecordToFieldsMap.ini file to the following directory:

  <Secure Agent installation directory>\apps\Data_Integration_Server\ext\deploy_to_main\bin\rdtm\extra\reserved\userfiles\netsuite

4. Update the file as required.

**Configuring the NetSuiteCustomFields.ini file**

You can specify custom NetSuite fields in the **Record Custom Fields** connection property when you configure a NetSuite connection so that the fields are available in Data Integration. Alternatively, if the NetSuite connection uses a Secure Agent group as the runtime environment, you can specify the custom NetSuite fields in the **NetSuiteCustomFields.ini file**.

If you choose to configure the NetSuiteCustomFields.ini file, perform the following steps:

1. Make a copy of the NetSuiteCustomFields.ini file, located in the following directory:

   <Secure Agent installation directory>\downloads\packagenetsuiteconnector.<version>\package\rdtm\javalib

2. Use the following guidelines to change the file to include custom NetSuite fields:
   - Create a separate section for each NetSuite record for which you want to add custom fields.
   - Add the custom fields using the following format, where the value of scriptId is the ID field in the NetSuite user interface for each custom field:

     ![Custom Field Format Example](image)

     For example:

     ![Custom Field Example](image)

   - Add the custom fields for NetSuite advanced search using the following format, where the value of scriptId is the ID field in the NetSuite user interface for each custom field:

     ![Advanced Search Field Format Example](image)

     For example:

     ![Advanced Search Field Example](image)
• If you want to read or write custom segment data, use the following format to add the custom segment fields:

```
<Object Name>
  custSegScriptIds=custseg1:select,custseg2:multiselect,custseg3:select....
```

where the value of scriptId is the ID field in the NetSuite user interface for each custom segment field.

For example:

```
[Employee]
  custSegScriptIds=custbody18:select,custbody19:multiselect,custbody20:select....
```

If you want to read data from or write data to child record custom segments, use the following format to add the child custom segment fields:

```
<Object Name>
  custSegScriptIds=custseg1:select,custseg2:multiselect,custseg3:select....
```

For example:

- [JournalEntry]
  custSegScriptIds=custbody_cseg1:select,custbody_cseg2:select,custbody_cseg3:select
- [JournalEntryLineList]
  custSegScriptIds=custcol_cseg1:select,custcol_cseg2:select,custcol_cseg3:select

3. Save the NetSuiteCustomFields.ini in the following location:

```
<Secure Agent installation directory>\apps\Data_Integration_Server\ext\deploy_to_main\bin\rdtm-extra\reserved\userfiles\netsuite
```

**Note:** The custom segment fields, which you add either in NetSuiteCustomFields.ini or connection properties, are not validated. For example, the metadata and duplicate field names.

## Configuring the NetSuiteSavedSearchFields.ini file

You can specify saved search fields in the **Saved Search Record Fields** connection property when you configure a NetSuite connection to ensure that all saved search fields are available in Data Integration. Alternatively, if the NetSuite connection uses a Secure Agent group as the runtime environment, you can configure the NetSuiteSavedSearchFields.ini file.

If you choose to configure the NetSuiteSavedSearchFields.ini file, perform the following steps:

1. Make a copy of the NetSuiteSavedSearchFields.ini file, located in the following directory:

```
<Secure Agent installation directory>\downloads\packageNetsuiteConnector.<version>\package\rdtm\javalib
```

2. Use the following guidelines to change the file to include all of the saved search fields:

   - Create a separate section for each NetSuite saved search record for which you want to add a saved search field, identified by a unique scriptId.

   Add the search fields using the following format:

```
[SavedSearchScriptIdToFieldsMap<version>]
  <savedSearchId>='<savedSearchDeclaredField1Name>,<savedSearchDeclaredField2Name>,<savedSearchCustomFieldScriptId1>,<savedSearchCustomFieldScriptId2>,<StandardJoin>
  <FieldName1>,
  customSearchJoin|<scriptId1>
```

The savedSearchId1 is ID field in the NetSuite user interface that specify the saved search record. The savedSearchDeclaredField1Name and savedSearchDeclaredField2Name are standard field names in the NetSuite user interface. The savedSearchCustomFieldScriptId1 and
savedSearchCustomFieldScriptId2 are ID fields in the NetSuite user interface for custom fields. The StandardJoin|FieldName1 is the standard join field and customSearchJoin|scriptId1 is the custom join field.

For example:

```plaintext
[SavedSearchScriptIdToFieldsMap]
1000=phone,email,custentity78,custentity65,userJoin|email,customSearchJoin|custrecord1424
```

In the example, 1000 is the saved Search ID, phone and email are standard field names, custentity78 and custentity65 are the script IDs of a custom field. userJoin|email is the standard join field and customSearchJoin|custrecord1424 is the custom join field script ID.

**Note:** When you use custom join, script ID appears in the Data Integration user interface.

- If you want to read custom segment data, use the following format to add the search custom segment fields:

```plaintext
[SavedSearchScriptIdToCustSegFieldsMap]
savedSearchId1 = custseg1:select,custseg2:multiselect,custseg3:select....
```

where custom segment field scriptId is appended to field type (select or multiselect) with ":" delimiter.

For example:

```plaintext
[SavedSearchScriptIdToCustSegFieldsMap]
741=custseg1:select,custentity_cseg1:select,custentity_csegcs_multsel:multiselect
```

3. Save the NetSuiteSavedSearchFields.ini in the following location:

```plaintext
<Secure Agent installation directory>\apps\Data_Integration_Server\ext\deploy_to_main\bin\rdtm-extra\reserved\userfiles\netsuite
```

### Fetching field name for saved search with standard join

Perform the following steps to fetch field name for standard join:


2. Under **Schema Browser**, select the object.

3. Under **Related Searches**, select the search object.

4. Under **Fields**, select the type for the standard join.

   You will find the name of the standard join field.

   The syntax for saved search INI file with standard join is as follows,

   ```plaintext
   [SavedSearchScriptIdToFieldsMap]<savedSearchId1>=<StandardJoin>|<FieldName1>,
   <StandardJoin>|<FieldName2>
   ```

   For example,

   ```plaintext
   [SavedSearchScriptIdToFieldsMap]
   290=usrjjoin|accountNumber, contactJoin|address
   ```

### Fetching script ID for saved search with custom join

Perform the following steps to fetch script ID for custom join:

1. Log in to your NetSuite account.

2. Select **Customization > Record Types**.

3. Select the custom object to get the script ID for the custom join field.
The syntax for saved search INI file with custom join is as follows:

```
[SavedSearchScriptIdToFieldsMap|savedSearchId1]= customSearchJoin|<scriptId1>,customSearchJoin|<scriptId2>
```

For example,

```
[SavedSearchScriptIdToFieldsMap]
290= customSearchJoin|custrecord1423,custSearchJoin|custrecord1421
```

### Reading custom record standard fields with custom join

To read custom record standard fields with custom join, configure the `DumpSavedSearchMetadata` property.

1. Go to **Administrator > Runtime Environments**.
   The Runtime Environments page appears.
2. Select the Secure Agent for which you want to set the custom configuration flag.
3. Click **Edit Secure Agent** in **Actions**.
   The Edit Secure Agent page appears.
4. Select the **Service** as **Data Integration Server** in the **Custom Configuration Details** section.
5. Select the **Type** as **Tomcat** in the **Custom Configuration Details** section.
6. Add **DumpSavedSearchMetadata** in the **Name** field.
7. Specify a file path in the **Value** field where you want to generate the **DumpSavedSearchMetadata file**.
   For example, `C:\\Dump.txt`. The path should contain `\\`.
   The following image shows the Custom Configuration Details section.

   ![Custom Configuration Details](image)

   **Note:** If the path contains only the file name, then the Secure Agent generates the **DumpSavedSearchMetadata file** in the following folder: `<Secure Agent installation directory>\apps \Data_Integration_Server\DIS Version\ICS\main\tomcat`. **If the path is not valid, the Secure Agent does not generate the DumpSavedSearchMetadata file.**
8. Click **OK**.
9. Create a task with saved search object to read custom record standard fields with custom join.

   When you save or run the task, the Secure Agent generates the **DumpSavedSearchMetadata file** and writes the metadata to this file. The Secure Agent fetches the metadata for the custom record standard fields in the following format:

   ```xml
   <script Id of custom record>_<standard search column name1>_CSJ,
   <script Id of custom record>_<standard search column name2>_CSJ
   ```

   For example:

   ```ini
   custbody_cseg1__internalId__CSJ,custbody_cseg1__name__CSJ
   ```

   **Note:** When you specify a field name, ensure that the length of the field name does not exceed 64 characters.

   If you want to override the metadata, you can copy the metadata in the **NetSuiteSavedSearchFields.ini file** and modify the metadata accordingly. Use the following format to add the search custom record standard fields in the **NetSuiteSavedSearchFields.ini file**:

   ```ini
   [SavedSearchScriptIdToFieldsMap|version>]
   <savedSearchId1>|CustomSearchJoin|<scriptId of custom record>_<standard field name>
   ```
For example:

[SavedSearchScriptIdToFieldsMap]
356=CustomSearchJoin|uss_custom_code__internalId

Alternatively, you can specify saved search fields in the Saved Search Record Fields connection property when you configure a NetSuite connection.

Internalid and externalid in saved search

When you create a new task or edit or refresh an existing task to read data from NetSuite saved search that contains an internalid or externalid field, the agent does not drill down the internalid or externalid field if you do not specify the saved search fields in the NetSuiteSavedSearchFields.ini file or Saved Search Record Fields connection property. The agent writes the internalid field as internalid and externalid field as externalid.

If you specify the saved search fields in the NetSuiteSavedSearchFields.ini file or Saved Search Record Fields connection property, the agent drills down the internalid or externalid field. The agent drills down the internalid field as internalid_internalid, internalid_externalid, internalid_type, and internalid_name. The agent drills down the externalid field as externalid_externalid, externalid_internalid, externalid_type, and externalid_name.

Note: If you edit or refresh an existing task, you must remap the internalid or externalid field or create a new target file.

Increasing heap size for concurrency in synchronization tasks

Users can configure synchronization tasks to use concurrent threads. If the NetSuite connection for synchronization tasks uses a Secure Agent group as the runtime environment, you can adjust heap size to optimize performance for concurrency.

You can adjust heap size in the Edit Secure Agent page or you can edit the pmrdtm configuration file.

Increasing heap size in the Edit Secure Agent page

You can adjust heap size in the Edit Secure Agent page to optimize performance for concurrency.

1. Click Administrator > Runtime Environments.
2. On the Runtime Environments page, if necessary, expand the Secure Agent groups to see the list of Secure Agents. Click the Secure Agent name from the list.
3. Click Edit Secure Agent in Actions.
   The Edit Secure Agent page appears.
4. In the System Configuration Details section, for Type, select DTM. Modify the JVMOption parameter to specify the heap size.
   For example, you might want to adjust the heap size to 512 MB if the concurrency thread setting is 10.

The following example shows the heap size set to 512 MB:

```
JVMOption1=-Xmx512m
JVMOption5=-XX:+HeapDumpOnOutOfMemoryError
JVMOption4=-Xloggc:/jvm_heap_stats_Clou_App.log
JVMOption3=-XX:+PrintGCDetails
JVMOption2=-XX:+PrintGCTimeStamps
JVMOption1=-verbose:gc
```
Handling parent and child or sibling objects with similar column name

If the parent object and the child or sibling object have similar column names, data from parent column is written to the parent column, and data from child or sibling column is written to the child or sibling column. You can set the UseJumbledDataForFieldsWithSameName property to control this behavior.

If you set the UseJumbledDataForFieldsWithSameName property to true, data from parent column is written to the child or sibling column, and data from child or sibling column is written to the parent column.

If you set the UseJumbledDataForFieldsWithSameName property to false, data from parent column is written to the parent column, and data from child or sibling column is written to the child or sibling column.

1. Go to Administrator > Runtime Environments.
2. On the Runtime Environments page, if necessary, expand the Secure Agent groups to see the list of Secure Agents. Click the Secure Agent name from the list.
3. Click Edit Secure Agent in Actions. The Edit Secure Agent page appears.
4. In the Custom Configuration Details section, for Type, select DTM.
5. Set the UseJumbledDataForFieldsWithSameName parameter to true or false as shown in the following image:

![Image](image-url)

6. Restart the Secure Agent.
NetSuite connections overview

Create a NetSuite connection to access NetSuite data. You can use NetSuite connections to specify sources, lookups, and targets in synchronization tasks, PowerCenter tasks, mappings, and mapping tasks. When you create a NetSuite connection, you enter properties specific to NetSuite.

You create a NetSuite connection on the Connections page, in the Mapping Designer when you create a mapping, or in the Synchronization task wizard, the PowerCenter Task Details page, or the Mapping task wizard when you create a task. The connection becomes available to the entire organization to use.

NetSuite token-based authentication

You can configure NetSuite connections to use token-based authentication to access NetSuite.

Token-based authentication is the preferred method to access NetSuite. When a connection uses token-based authentication, the agent uses a token ID and token secret to access NetSuite instead of a user name and password. The NetSuite connection must use WSDL URL version 2015_2 and above to use token-based authentication.

To set up token-based authentication, you install a NetSuite bundle for Data Integration token-based authentication and generate a token ID and token secret in NetSuite. The token does not expire unless you revoke it from your NetSuite account. However, you might need to update the bundle and generate a new token if Informatica updates the bundle version in the future.

You enter the generated token ID and token secret when you specify the NetSuite connection properties in Data Integration. If you include token information and a user name and password for a connection that uses WSDL URL version 2015_2, the agent uses token-based authentication to access NetSuite.
Generating a NetSuite token ID and token secret

To use token-based authentication, in NetSuite, you install an Informatica token-based authentication bundle and generate the token ID and token secret. You use the token ID and token secret when you configure the NetSuite connection in Data Integration.

1. Log in to NetSuite using a Full Access or Administrator account.
2. Navigate to Customization > SuiteBundler > Search and Install Bundles.
3. Search for the keyword, “InformaticaTBABundle”.
   A bundle with Bundle ID of 116143 appears in the search results.
4. Select InformaticaTBABundle and install it.
6. For Application Name, select InformaticaTBAIntegration.
7. Write down the access token and token secret displayed on the page.

You enter the token ID and token secret in Data Integration when you configure the NetSuite connection.

Note: If you lose the token information, you must generate another token in NetSuite. NetSuite does not provide token information for previously generated tokens.

NetSuite connection properties

When you set up a NetSuite connection, you must configure the connection properties.

The following table describes the NetSuite connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runtime Environment</td>
<td>The name of the runtime environment where you want to run the tasks.</td>
</tr>
<tr>
<td>Token ID</td>
<td>The token ID generated in NetSuite. Required if you want to use token-based authentication to access NetSuite. Optional if you use a user name and password to access NetSuite.</td>
</tr>
<tr>
<td>Token Secret</td>
<td>The token secret generated in NetSuite. Required if you want to use token-based authentication to access NetSuite. Optional if you use a user name and password to access NetSuite.</td>
</tr>
<tr>
<td>Username</td>
<td>User name for a NetSuite account. User name is an email address. Optional if you use token-based authentication to access NetSuite.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the NetSuite account. Optional if you use token-based authentication to access NetSuite.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Service URL      | NetSuite WSDL URL. From version 2019_2 of the NetSuite WSDL URL, you can enter the WSDL URL used by your NetSuite account instead of the default service URL.  
The service URL used by the NetSuite account is in the following format:  
<NetSuite_account_URL>/wsdl/v2019_2_0/netsuite.wsdl.  
Informatica recommends that you use the WSDL URL specific to the NetSuite account that you use.  
For more information, see "Configuring NetSuite account-specific service URL" on page 20.  
The default service URL is https://webservices.netsuite.com/wsdl/v2019_2_0/netsuite.wsdl. |
| Account          | NetSuite account ID. To find your account ID, log in to NetSuite and click Setup > Integration > Web Services Preferences.  
If you cannot access the Setup menu, navigate to Support > Go to Suite Answers > Contact support by phone. The page will display your account ID. |
| Application ID   | NetSuite application ID.  
Required when you use WSDL URL version 2015_2 or above without token-based authentication. If the application ID property is blank, the agent uses the Informatica application ID.  
To find your application ID, log in to NetSuite and click Setup > Integration > Manage Integrations. If you do not have an application ID, you can create one. On the Manage Integrations page, click New.  
After you save the application ID, you can view the application ID number on the Manage Integrations page. |
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Record Custom Fields | Specify custom NetSuite fields so that they are available in Data Integration. Add the custom fields using the following format, where the value of scriptId is the ID field in the NetSuite user interface for each custom field:  
  `<Object Name>scripts`  
  scriptIds = `<custom field name1>`, `<custom field name2>`, `<custom field name3>`  
  For example:  
  `[Sales]scripts`  
  scriptIds = `discountPrice`, `salesDescription`, `salesEvent3`  
  Add the custom fields for NetSuite advanced search using the following format, where the value of scriptId is the ID field in the NetSuite user interface for each custom field:  
  `<Object Name>scripts`  
  scriptIds = `<custom field name1>`, `<custom field name2>`, `<custom field name3>`  
  For example:  
  `[EmployeeSearchAdvanced]scripts`  
  scriptIds = `custentity74`, `custentity66`  
  If you want to read or write custom segment data, use the following format to add the custom segment fields:  
  `<Object Name>custSegScripts`  
  custSegScriptIds = `custseg1:select`, `custseg2:multiselect`, `custseg3:select`...  
  where the value of scriptId is the ID field in the NetSuite user interface for each custom segment field.  
  For example,  
  `[Employee]custSegScripts`  
  custSegScriptIds = `custentity_cseg1:select`, `custentity_csegcs_multisel:multiselect`  
  If you want to read data from or write data to child record custom segments, use the following format to add the child custom segment fields:  
  `<Object Name>custSegScripts`  
  custSegScriptIds = `custseg1:select`, `custseg2:multiselect`, `custseg3:select`...  
  For example:  
  `[JournalEntry]custSegScripts`  
  custSegScriptIds = `custbody_cseg1:select`, `custbody_cseg2:select`, `custbody_cseg3:select`  
  `[JournalEntryLineList]custSegScripts`  
  custSegScriptIds = `custcol_cseg1:select`, `custcol_cseg2:select`, `custcol_cseg3:select`
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Record Filter Fields</td>
<td>Map NetSuite record field names with related NetSuite search record field names so that you can use the fields in filters.</td>
</tr>
<tr>
<td></td>
<td>List the record field names and related SearchBasic field names, as follows:</td>
</tr>
<tr>
<td></td>
<td>[record 1]</td>
</tr>
<tr>
<td></td>
<td>&lt;record field name&gt;=&lt;SearchBasic field name&gt;&lt;record field name2&gt;=&lt;SearchBasic field name2&gt;</td>
</tr>
<tr>
<td></td>
<td>[record 2]</td>
</tr>
<tr>
<td></td>
<td>&lt;record field name&gt;=&lt;SearchBasic field name&gt;&lt;record field name2&gt;=&lt;SearchBasic field name2&gt;&lt;record field name3&gt;=&lt;SearchBasic field name3&gt;</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>[Account]</td>
</tr>
<tr>
<td></td>
<td>acctName=nameaddr1=address1</td>
</tr>
<tr>
<td></td>
<td>To read transactional data from NetSuite when memorized transaction is enabled in the NetSuite account, add the record field names and related SearchBasic field name in the following format:</td>
</tr>
<tr>
<td></td>
<td>[record 1]</td>
</tr>
<tr>
<td></td>
<td>&lt;record field name&gt;=&lt;SearchBasic field name&gt;</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>[JournalEntry]</td>
</tr>
<tr>
<td></td>
<td>reversalEntry=memorized</td>
</tr>
<tr>
<td>Saved Search Record Fields</td>
<td>Saved search fields in NetSuite do not appear in Data Integration if they have a null value, by default. Add saved search fields to ensure that saved search fields are available in Data Integration even when they have a null value. Create a separate section for each NetSuite saved search record for which you want to add a saved search field, identified by a unique scriptId.</td>
</tr>
<tr>
<td></td>
<td>Add the search fields using the following format:</td>
</tr>
<tr>
<td></td>
<td>&lt;savedSearchId1&gt;=&lt;savedSearchDeclaredFieldName1Name&gt;,&lt;savedSearchDeclaredFieldName2Name&gt;,&lt;savedSearchCustomFieldScriptId1&gt;,&lt;savedSearchCustomFieldScriptId2&gt;,&lt;StandardJoin&gt;</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>1000=phone,email,custentity78,custentity65,userJoin</td>
</tr>
<tr>
<td></td>
<td>If you want to read custom segment data, use the following format to add the search custom segment fields:</td>
</tr>
<tr>
<td></td>
<td>[savedSearchId1]=custseg1:select,custseg2:multiselect,custseg3:select...</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>[741]=custseg1:select,custentity_cseg1:select,custentity_csegcs_multsel:multiselect</td>
</tr>
<tr>
<td></td>
<td>If you want to override the metadata of a task, which is created to read custom record standard fields with custom join, use the following format to add the search custom record standard fields:</td>
</tr>
<tr>
<td></td>
<td>&lt;savedSearchId1&gt;&gt;&lt;CustomSearchJoin</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>356=CustomSearchJoin</td>
</tr>
</tbody>
</table>
NetSuite WSDL URL versions

NetSuite connections use the NetSuite Web Service Description Language (WSDL) URL to access NetSuite metadata and NetSuite data.

By default, NetSuite connections use version 2019_2_0 of the NetSuite WSDL URL as shown in the following URL:

https://webservices.netuite.com/wsd1/v2019_2_0/netuite.wsdl

You can also use version 2019_1_0 of the NetSuite WSDL URL as shown in the following URL:

https://webservices.netuite.com/wsd1/v2019_1_0/netuite.wsdl

If your NetSuite account does not use one of these WSDL URL versions, update the WSDL URL for your NetSuite account.

If you have a NetSuite sandbox account, you can use the following NetSuite WSDL URL to access Data Integration in the sandbox:

https://webservices.sandbox.netuite.com/wsd1/v2019_2_0/netuite.wsdl

Note: NetSuite Connector is certified to work with 2019_1 and 2019_2 API endpoints of the NetSuite WSDL URL. NetSuite Connector works with earlier versions of NetSuite WSDL URL at runtime. NetSuite Connector with version 2019_2 of the NetSuite WSDL URL is certified to work with NetSuite 2020_1 GA release.

Rules and guidelines for NetSuite connections

Consider the following rules and guidelines for NetSuite connections:

- When you select a connection in a mapping, synchronization task, or mapping task wizard, you can search for the object or objects that you want to use. You can search for objects using the name, label, description, or type parameter.

- Connections display business names for field names instead of technical names by default. You can configure tasks to display technical names instead of business names with the Display technical names instead of labels option.

- You need a separate license for each connection to the same NetSuite account that a task makes. For example, to use the same NetSuite account as source, lookup, and target in a task, you need three NetSuite licenses.

- You can use multiple concurrency threads to improve the performance of a task when you use NetSuite. Informatica recommends that you use token-based authentication when you have a basic NetSuite account to improve the performance.

  Note: For web services that use request-level credentials for authentication, the governance limit for concurrent requests is set to 1 for a basic NetSuite account (without SuiteCloud Plus License). For web services that use token-based authentication, the limit for concurrent requests is set to 5 for a basic NetSuite account (without SuiteCloud Plus License).

Configuring NetSuite account-specific service URL

Perform the following steps to use your NetSuite account-specific service URL:

1. Log in to NetSuite and click Setup > Company > Company Information.
2. On the Company Information page, click Company URLs.
The **SUITE TALK (SOAP AND REST WEB SERVICES)** field displays the account-specific URL in the following format:

https://<NetSuite_account_ID>.suitetalk.api.netsuite.com

3. Copy and paste the account-specific URL from step 2 in the following format in the service URL:

https://<Net_suite_account_ID>.suitetalk.api.netsuite.com/wsdl/v2019_2_0/netsuite.wsdl

---

**Troubleshooting a NetSuite connection**

When you create a NetSuite connection, the following error can occur:

Test Connection Failed for <connection name>. ConnectionFailedException: [Connection]. ExceededRequestLimitFault: Only one request may be made against a session at a time.

You might receive this message because you can use no more than one NetSuite connection at a time. To resolve the issue, you can request the Suite Cloud Plus account from NetSuite, which allows up to 10 mappings for each user.
Synchronization tasks with NetSuite

This chapter includes the following topics:

- NetSuite sources in synchronization tasks, 22
- NetSuite targets in synchronization tasks, 23
- NetSuite lookups in synchronization tasks, 23
- Synchronization task schedule and advanced options for NetSuite, 24
- Rules and guidelines for NetSuite objects in a synchronization task, 25

NetSuite sources in synchronization tasks

You can use a NetSuite object as a single source in a synchronization task. You can also use multiple related NetSuite standard objects as sources in a synchronization task.

When you use multiple NetSuite source objects, you can select a standard object as the primary source, then you add one or more sibling objects or a single child object. When you select a list/record type object, its related object appears in the sibling list. NetSuite does not support the use of child and sibling objects at the same time. You can use a custom or saved search NetSuite object as a single source.

You configure NetSuite source properties on the Source page of the Synchronization Task wizard.

The following table describes the NetSuite source properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Name of the source connection.</td>
</tr>
<tr>
<td>Source Type</td>
<td>Select Single or Multiple.</td>
</tr>
<tr>
<td>Source Object</td>
<td>For a single source. Select the source object.</td>
</tr>
<tr>
<td>Add Primary</td>
<td>For multiple sources. Select the primary source object.</td>
</tr>
<tr>
<td>Add Child</td>
<td>For multiple sources. Displays child objects related to the selected source object. Select a single child object.</td>
</tr>
</tbody>
</table>
You can use a NetSuite object as a single target in a synchronization task. You can also use multiple related NetSuite standard objects as targets in a synchronization task.

You can use a pair of parent and child standard objects as targets. When you configure two NetSuite targets, you can select a standard object as the parent object and add a child object. You can use a custom object as a single target.

You configure NetSuite target properties on the **Target** page of the Synchronization Task wizard.

The following table describes the NetSuite target properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Name of the target connection.</td>
</tr>
<tr>
<td>Target Object</td>
<td>Select the primary target object.</td>
</tr>
<tr>
<td>Child Object</td>
<td>If the primary target object is a standard object, select a related child object to use two objects as the target.</td>
</tr>
<tr>
<td>Display technical names instead of labels</td>
<td>Displays technical names instead of business names.</td>
</tr>
<tr>
<td>Display target fields in alphabetical order</td>
<td>Displays target fields in alphabetical order instead of the order returned by the source system.</td>
</tr>
</tbody>
</table>

**NetSuite lookups in synchronization tasks**

When you configure field mappings in a synchronization task, you can create a lookup to a NetSuite standard or custom object.

When you use a NetSuite object as a lookup, you do not need to configure specific NetSuite properties. You can use a custom field in a lookup condition if the field is filterable.
Synchronization task schedule and advanced options for NetSuite

When you configure a synchronization task to use a NetSuite source or target, you configure advanced properties on the Schedule page of the Synchronization Task wizard.

The following table describes the NetSuite advanced source properties:

<table>
<thead>
<tr>
<th>Advanced Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Date</td>
<td>Start date and time to capture the changed data. Use the following format: YYYY-MM-DD'T'hh:mm:ss.SSSSZ. To specify the last date or the last date and time when the task ran successfully, enter the $LastRunDate or $LastRunTime data filter variables. To perform a full read, do not use the Start Date and End Date properties.</td>
</tr>
<tr>
<td>End Date</td>
<td>End date and time to capture the changed data. Use the following format: YYYY-MM-DD'T'hh:mm:ss.SSSSZ. To specify the last date or the last date and time when the task ran successfully, enter the $LastRunDate or $LastRunTime data filter variables. To perform a full read, do not use the Start Date and End Date properties.</td>
</tr>
<tr>
<td>Page Size</td>
<td>Number of rows that the Secure Agent fetches per page. Default is 100 rows. The maximum value is 1000.</td>
</tr>
<tr>
<td>Get Deleted</td>
<td>Includes records that were deleted between the start date and end date for changed data capture.</td>
</tr>
<tr>
<td>Concurrent Threads</td>
<td>Number of concurrent threads. To run a synchronization task that uses concurrent threads instead of sequential threads, specify the number of concurrent threads allowed. To use concurrent threads for synchronization tasks, your NetSuite account must be concurrency-enabled. To optimize performance results with concurrent threads, your administrator might need to adjust the heap size. For example, a heap size of 512 MB for 10 threads might optimize performance results. <strong>Note:</strong> You cannot use a saved search as a source object in a synchronization task that uses concurrent threads. The default value is 1.</td>
</tr>
<tr>
<td>Maximum Number of Records to Be Read</td>
<td>Maximum number of records read from the source. For example, a value of 100 means that the agent reads 100 records from the source. The default value is 0, which means the agent reads all records.</td>
</tr>
<tr>
<td>Number of Retry</td>
<td>Number of times the agent attempts to execute the request. Default value is 0. <strong>Note:</strong> Retry mechanism works after the transaction starts.</td>
</tr>
<tr>
<td>Retry Delay</td>
<td>Number of seconds the agent waits before it executes the request again. Default value is 200. The agent reconnects to the NetSuite port every time it executes the request and send the existing search ID to the NetSuite port.</td>
</tr>
</tbody>
</table>
The following table describes the NetSuite advanced target properties:

<table>
<thead>
<tr>
<th>Advanced Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replace All</td>
<td>Removes existing data from the child object before writing new data to the target.</td>
</tr>
<tr>
<td>Batch Size</td>
<td>Number of rows that the Secure Agent writes in a batch to the target. When the batch size is 0, the Secure Agent writes data to the target one row at a time. When the batch size is greater than 0, the Secure Agent writes data to the target in batches of the specified size. The default value is 100. For insert and delete operations, the maximum value is 200. For upsert and update operations, the maximum value is 100.</td>
</tr>
<tr>
<td>Success File</td>
<td>Name of the file that contains rows successfully written to the target. The agent writes success log files to the following directory: <code>&lt;Secure Agent installation directory&gt;/apps/Data_Integration_Server/data _success</code></td>
</tr>
<tr>
<td>Reject File</td>
<td>Name of the file that contains rows that were not written to the target. The agent writes error log files to the following directory: <code>&lt;Secure Agent installation directory&gt;/apps/Data_Integration_Server/data _error</code></td>
</tr>
<tr>
<td>Concurrent Threads</td>
<td>Number of concurrent threads. To run a synchronization task that uses concurrent threads instead of sequential threads, specify the number of concurrent threads allowed. To use concurrent threads for synchronization tasks, your NetSuite account must be concurrency-enabled. To optimize performance results with concurrent threads, your administrator might need to adjust the heap size. For example, a heap size of 512 MB for 10 threads might optimize performance results. The default value is 1.</td>
</tr>
</tbody>
</table>

Rules and guidelines for NetSuite objects in a synchronization task

Consider the following rules and guidelines for NetSuite objects used as sources, targets, and lookups in synchronization tasks:

- Data Integration supports NetSuite Entity fields, Item fields, CRM fields, Transaction Body/Column/Item fields, and other custom fields. Item Number data is not supported.
- NetSuite does not support multiple filters on a field. If you configure multiple filters for a field, the agent uses the last-defined filter for the field.
- You cannot perform a NetSuite search on customized fields that reference another record.
- Filters for NetSuite Multi-Select and Standard Record custom fields are not supported.
- You cannot use advanced data filters for NetSuite sources in synchronization tasks.
- You cannot use parameters defined in a parameter file in data filters for NetSuite sources in synchronization tasks.
- You can remove mapped source fields from a NetSuite source. The synchronization task does not fail as a result.
• The WSDL version 2014_2 does not include the isBookSpecific column for new book account objects. Consequently, you cannot see the isBookSpecific column with connections that use WSDL version 2014_2.

• NetSuite returns both the date and time using the following format for all date or time data types: yyyy-mm-ddThh:mm:ss <AM/PM>. For the TimeOfDay data type, the date defaults to 1970-01-01.

• Field metadata information such as primary key or not-null does not display in Data Integration.

• When you include custom fields of NetSuite custom objects in a task, they are added to the field list of the child record rather than the parent record. For example, transaction column fields and transaction item options.

• Metadata fetch does not work when a saved search record contains custom fields and multi-select standard join fields.

• When your NetSuite connection uses SuiteTalk and the BodyOnlyFields parameter is set to false, the Campaign object cannot be read.

• Data Integration does not support NetSuite formula fields.

• Due to a NetSuite limitation, if you use a NetSuite connection that uses WSDL 18.1, the task fetches the value of the Language field as ENGLISH_US. The task fetches the value of the Language field as US_ENGLISH if you use a NetSuite connection that uses WSDL 17.2 or earlier version.

• When you use a NetSuite connection, which contains saved search record fields in Connection attribute, to create a task to read from NetSuite saved search, the agent appends _CSJ to custom fields. Also, when you edit or refresh an existing task that reads from NetSuite saved search, the agent appends _CSJ to custom fields.

• If you edit or refresh an existing task to read from a saved search object that has an internalId or externalId field, the task might fail with the following error message:

Transformation stopped due to a fatal error in the mapping. The expression [internalId_InternalId] contains the following errors [<<PM Parse Error>> [internalId_InternalId]: invalid symbol reference ... >>>internalId_InternalId<<]<].

You need to remap the appropriate fields in the existing task or create a new target file. For more information, see “InternalId and externalId in saved search” on page 13.
CHAPTER 4

Mappings and mapping tasks with NetSuite

This chapter includes the following topics:

- NetSuite objects in mappings, 27
- NetSuite objects in template-based mapping tasks, 31
- Rules and guidelines for NetSuite objects, 32

NetSuite objects in mappings

When you create a mapping, you can configure a Source, Target, or Lookup transformation to represent a NetSuite object.

NetSuite sources in mappings

In a mapping, you can configure a Source transformation to represent a single NetSuite source or multiple NetSuite sources.

You can use multiple related NetSuite standard objects as a source. You can select a standard object as the primary source, then you add one or more sibling objects or a single child object. When you select a list/record type object, its related object appears in the sibling list. NetSuite does not support the use of child and sibling objects at the same time.

The following table describes the NetSuite source properties that you can configure in a Source transformation:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Name of the source connection.</td>
</tr>
<tr>
<td>Source type</td>
<td>Select Single Object or Multiple Objects.</td>
</tr>
<tr>
<td>Object</td>
<td>For a single source. Select the source object.</td>
</tr>
<tr>
<td>Add Source Object</td>
<td>For multiple sources. Select the primary object.</td>
</tr>
<tr>
<td>Add Child Object</td>
<td>For multiple sources. Select the desired child object.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Add Sibling Objects</td>
<td>For multiple sources. Select the desired sibling objects.</td>
</tr>
<tr>
<td>Filter</td>
<td>Filters value in a read operation. You can add conditions to filter records and reduce the number of rows that the Secure Agent reads from the source.</td>
</tr>
<tr>
<td>Sort</td>
<td>Not supported.</td>
</tr>
</tbody>
</table>

The following table describes the NetSuite source advanced properties that you can configure in a Source transformation:

<table>
<thead>
<tr>
<th>Advanced Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Date</td>
<td>Start date and time to capture the changed data. Use the following format: YYYY-MM-DD'T'hh:mm:ss.SSSSZ. To perform a full read, do not use the Start Date and End Date properties.</td>
</tr>
<tr>
<td>End Date</td>
<td>End date and time to capture the changed data. Use the following format: YYYY-MM-DD'T'hh:mm:ss.SSSSZ. To perform a full read, do not use the Start Date and End Date properties.</td>
</tr>
<tr>
<td>Page Size</td>
<td>Number of rows that the Secure Agent fetches per page. Default is 100 rows. The maximum value is 1000.</td>
</tr>
<tr>
<td>Get Deleted</td>
<td>Includes records that were deleted between the start date and end date for changed data capture.</td>
</tr>
<tr>
<td>Concurrent Threads</td>
<td>Number of concurrent threads. Specify the number of concurrent threads allowed. To use concurrent threads for mapping tasks, your NetSuite account must be concurrency-enabled. To optimize performance results with concurrent threads, your administrator might need to adjust the heap size. For example, a heap size of 512 MB for 10 threads might optimize performance results. Default value is 1. You cannot use a saved search as a source object in a mapping task that uses concurrent threads.</td>
</tr>
<tr>
<td>Maximum Number of Records to Be Read</td>
<td>Maximum number of records read from the source. For example, a value of 100 means that the agent reads 100 records from the source. The default value is 0, which means the agent reads all records.</td>
</tr>
<tr>
<td>Number of Retry</td>
<td>Number of times the agent attempts to execute the request. Default value is 0. Note: Retry mechanism works after the transaction starts.</td>
</tr>
<tr>
<td>Retry Delay</td>
<td>Number of seconds the agent waits before it executes the request again. Default value is 200. The agent reconnects to the NetSuite port every time it executes the request and send the existing search ID to the NetSuite port.</td>
</tr>
</tbody>
</table>
Adding multiple source objects

To use multiple source objects in a Source transformation, you select the primary source object and the child or sibling objects on the Source tab of the Properties panel.

1. In the Objects and Relationships section, click the arrow to open the Action menu and then select Add Source Object.

2. Select the primary object.

3. Click the arrow in the primary object’s row and from the Related Object Actions menu, select either Add Child Object or Add Sibling Objects.

4. Select the desired child object or one or more sibling objects.

After you select the source objects, the Objects and Relationships section displays the source objects and the relationship between the objects, as shown in the following image:

NetSuite targets in mappings

In a mapping, you can configure a Target transformation to represent a single NetSuite target or multiple NetSuite targets.

When you use multiple NetSuite target objects, select a standard object as the primary target and then add a child object. If you use multiple NetSuite target objects, you must map at least one field in the parent target object and one field in the child target object. Also, the parent target object’s External ID is mandatory. You can use a custom object as a single target.

The following table describes the NetSuite target properties that you can configure in a Target transformation:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Name of the target connection.</td>
</tr>
<tr>
<td>Target Type</td>
<td>Select Single Object or Multiple Objects.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Object</td>
<td>Target object for a single target or primary target object for multiple targets.</td>
</tr>
<tr>
<td>Child Object</td>
<td>For multiple targets. Displays child objects related to the selected target object. Select a single child object.</td>
</tr>
<tr>
<td>Operation</td>
<td>Target operation.</td>
</tr>
</tbody>
</table>

The following table describes the NetSuite target advanced properties that you can configure in a Target transformation:

<table>
<thead>
<tr>
<th>Advanced Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replace All</td>
<td>Removes existing data from the child object before writing new data to the target.</td>
</tr>
<tr>
<td>Batch Size</td>
<td>Number of rows that the Secure Agent writes in a batch to the target. When the batch size is 0, the Secure Agent writes data to the target one row at a time. When the batch size is greater than 0, the Secure Agent writes data to the target in batches of the specified size. The default value is 100. For insert and delete operations, the maximum value is 200. For upsert and update operations, the maximum value is 100.</td>
</tr>
<tr>
<td>Success File</td>
<td>Name of the file that contains rows successfully written to the target.</td>
</tr>
<tr>
<td>Error File</td>
<td>Name of the file that contains rows that were not written to the target.</td>
</tr>
<tr>
<td>Concurrent Threads</td>
<td>Number of concurrent threads. Specify the number of concurrent threads allowed. To use concurrent threads for mapping tasks, your NetSuite account must be concurrency-enabled. To optimize performance results with concurrent threads, your administrator might need to adjust the heap size. For example, a heap size of 512 MB for 10 threads might optimize performance results. Default value is 1. You cannot use a saved search as a source object in a mapping task that uses concurrent threads.</td>
</tr>
<tr>
<td>Success File Directory</td>
<td>Directory for the success log files. Specify a directory path that is available on each agent machine in the runtime environment. By default, the agent writes the success log files to the following directory: <code>&lt;Secure Agent installation directory&gt;\apps\Data_Integration_Server\data\success</code></td>
</tr>
<tr>
<td>Error File Directory</td>
<td>Directory for the error log files. Specify a directory path that is available on each agent machine in the runtime environment. By default, the agent writes the error log files to the following directory: <code>&lt;Secure Agent installation directory&gt;\apps\Data_Integration_Server\data\error</code></td>
</tr>
<tr>
<td>Forward Rejected Rows</td>
<td>Determines whether the transformation passes rejected rows to the next transformation or drops rejected rows. By default, the agent forwards rejected rows to the next transformation.</td>
</tr>
</tbody>
</table>
NetSuite lookups in mappings

In a mapping, you can configure a Lookup transformation to represent a NetSuite object. You can use Netsuite standard objects and custom objects as lookups.

When you use a NetSuite object as a lookup, you do not need to configure specific NetSuite properties. You can use a custom field in a lookup condition if the field is filterable.

NetSuite objects in template-based mapping tasks

When you configure a mapping task based on an integration template, you can configure advanced properties for NetSuite sources and targets.

NetSuite sources in mapping tasks

For NetSuite source connections used in template-based mapping tasks, you can configure advanced properties in the Sources page of the Mapping Task wizard.

You can configure the following advanced properties:

<table>
<thead>
<tr>
<th>Advanced Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Date</td>
<td>Start date and time to capture the changed data. Use the following format: YYYY-MM-DD'T'hh:mm:ss.SSSZ. To perform a full read, do not use the Start Date and End Date properties.</td>
</tr>
<tr>
<td>End Date</td>
<td>End date and time to capture the changed data. Use the following format: YYYY-MM-DD'T'hh:mm:ss.SSSZ. To perform a full read, do not use the Start Date and End Date properties.</td>
</tr>
<tr>
<td>Page Size</td>
<td>Number of rows that the Secure Agent fetches per page. Default is 100 rows. The maximum value is 1000.</td>
</tr>
<tr>
<td>Get Deleted</td>
<td>Includes records that were deleted between the start date and end date for changed data capture.</td>
</tr>
</tbody>
</table>

NetSuite targets in mapping tasks

For NetSuite target connections used in template-based mapping tasks, you can configure advanced properties in the Targets page of the Mapping Task wizard.
You can configure the following advanced properties:

<table>
<thead>
<tr>
<th>Advanced Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replace All</td>
<td>Removes existing data from the child object before writing new data to the target.</td>
</tr>
<tr>
<td>Batch Size</td>
<td>Number of rows that the Secure Agent writes in a batch to the target. When the batch size is 0, the Secure Agent writes data to the target one row at a time. When the batch size is greater than 0, the Secure Agent writes data to the target in batches of the specified size. The default value is 100. For insert and delete operations, the maximum value is 200. For upsert and update operations, the maximum value is 100.</td>
</tr>
<tr>
<td>Success File</td>
<td>Name of the file that contains rows successfully written to the target.</td>
</tr>
<tr>
<td>Reject File</td>
<td>Name of the file that contains rows that were not written to the target.</td>
</tr>
<tr>
<td>Success File</td>
<td>Directory for the success log files. Specify a directory path that is available on each agent machine in the runtime environment. By default, the agent writes the success log files to the following directory: <code>&lt;Secure Agent installation directory&gt;\apps\Data_Integration_Server\data\success</code></td>
</tr>
<tr>
<td>Error File</td>
<td>Directory for the error log files. Specify a directory path that is available on each agent machine in the runtime environment. By default, the agent writes the error log files to the following directory: <code>&lt;Secure Agent installation directory&gt;\apps\Data_Integration_Server\data\error</code></td>
</tr>
</tbody>
</table>

**Rules and guidelines for NetSuite objects**

Consider the following rules and guidelines for NetSuite objects used as sources, targets, and lookups in mappings and mapping tasks:

- Data Integration supports NetSuite Entity fields, Item fields, CRM fields, Transaction Body/Column/Item fields, and other custom fields. Item Number data is not supported.
- NetSuite does not support multiple filters on a field. If you configure multiple filters for a field, the agent uses the last-defined filter for the field.
- You cannot perform a NetSuite search on customized fields that reference another record.
- Filters for NetSuite Multi-Select and Standard Record custom fields are not supported.
- The WSDL version 2014_2 does not include the isBookSpecific column for new book account objects. Consequently, you cannot see the isBookSpecific column with connections that use WSDL version 2014_2.
- NetSuite returns both the date and time using the following format for all date or time data types: yyyy-mm-ddThh:mm:ss <AM/PM>. For the TimeOfDay data type, the date defaults to 1970-01-01.
- Field metadata information such as primary key or not-null does not display in Data Integration.
- In an integration template, do not use data type link rules to connect NetSuite sources to source qualifier objects. Use the data type link rule between the source qualifier object and the rest of the data flow.
- To use multiple NetSuite sources in a custom integration task, use a NetSuite account that allows concurrent connections.
- When you include custom fields of NetSuite custom objects in a task, they are added to the field list of the child record rather than the parent record. For example, transaction column fields and transaction item options.
- Metadata fetch does not work when a saved search record contains custom fields and multi-select standard join fields.
- When your NetSuite connection uses SuiteTalk and the BodyOnlyFields parameter is set to false, the Campaign object cannot be read.
- Data Integration does not support NetSuite formula fields.
- Due to a NetSuite limitation, if you use a NetSuite connection that uses WSDL 18.1, the task fetches the value of the Language field as ENGLISH_US. The task fetches the value of the Language field as US_ENGLISH if you use a NetSuite connection that uses WSDL 17.2 or earlier version.
- When you use a Netsuite connection, which contains saved search record fields in Connection attribute, to create a task to read from NetSuite saved search, the agent appends _CSJ to custom fields. Also, when you edit or refresh an existing task that reads from NetSuite saved search, the agent appends _CSJ to custom fields.
- If you edit or refresh an existing task to read from a saved search object that has an internalId or externalId field, the task might fail with the following error message:

  Transformation stopped due to a fatal error in the mapping. The expression [internalId_InternalId] contains the following errors [<<FM Parse Error>> [internalId_InternalId]: invalid symbol reference ... >>>>internalId_InternalId<<<<<].

  You need to remap the appropriate fields in the existing task or create a new target file. For more information, see "Internalid and externalid in saved search" on page 13.
This appendix includes the following topics:

- **NetSuite data type reference overview, 34**
- **NetSuite and transformation data types, 34**

### NetSuite data type reference overview

Data Integration uses the following data types in mappings, synchronization tasks, and mapping tasks with NetSuite:

**NetSuite native data types**

NetSuite data types appear in the Fields tab for Source and Target transformations when you choose to edit metadata for the fields.

**Transformation data types**

Set of data types that appear in the remaining transformations. They are internal data types based on ANSI SQL-92 generic data types, which Data Integration uses to move data across platforms.

Transformation data types appear in all remaining transformations in a mapping, synchronization task, or mapping task.

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

### NetSuite and transformation data types

The following table lists the NetSuite data types that Data Integration supports and the corresponding transformation data types:

<table>
<thead>
<tr>
<th>NetSuite Native Data Type</th>
<th>Transformation Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Integer</td>
<td>Precision 5, scale 0</td>
</tr>
<tr>
<td>Check_box</td>
<td>Integer</td>
<td>Precision 10, scale 0</td>
</tr>
<tr>
<td>NetSuite Native Data Type</td>
<td>Transformation Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Currency</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Date</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)</td>
</tr>
<tr>
<td>Datetime</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)</td>
</tr>
<tr>
<td>Decimal_number</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Document</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Double</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>E-mail_address</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Free-form_text</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Hyperlink</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Inline_HTML</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Int</td>
<td>Integer</td>
<td>Precision 10, scale 0</td>
</tr>
<tr>
<td>Integer_number</td>
<td>Integer</td>
<td>Precision 10, scale 0</td>
</tr>
<tr>
<td>List/Record</td>
<td>String, Data/Time, Double, Integer</td>
<td>List/Record is treated as a source. Transformations convert List/Record fields to the appropriate transformation data type.</td>
</tr>
<tr>
<td>Long</td>
<td>Bigint</td>
<td>-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0</td>
</tr>
<tr>
<td>Multiple_select</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Password</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Percent</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
<tr>
<td>Phone_number</td>
<td>String</td>
<td>1 to 104,857,600 characters*</td>
</tr>
</tbody>
</table>
| RecordRef                | String                   | 1 to 104,857,600 characters*  
  When you include RecordRef in a mapping, it is converted into four String fields: Type, Internal ID, External ID, and Base Name. |
| Rich_text                | String                   | 1 to 104,857,600 characters* |
| String                   | String                   | 1 to 104,857,600 characters* |
| Text_area                | String                   | 1 to 104,857,600 characters* |
### NetSuite Native Data Type

<table>
<thead>
<tr>
<th>NetSuite Native Data Type</th>
<th>Transformation Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)</td>
</tr>
<tr>
<td>Time_of_day</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)</td>
</tr>
</tbody>
</table>

* String data types are imported with a precision of 256 characters. You can change the precision to the appropriate length. To improve performance, decrease the precision of String data types.

## List/Record data type

Data Integration treats the List/Record data type as a source. A list contains various fields. When you use a NetSuite source or target, Data Integration converts the data types of list fields to the appropriate transformation data types.

For example, if a list contains NetSuite Date and RecordRef columns, Data Integration converts these columns to transformation Date/Time and String data types, respectively.
INDEX

C
changed data capture
   guidelines for NetSuite 6
connections
   NetSuite 5, 16
   custom NetSuite fields 7, 16

D
data filters
   variables 22
data types
   List/Record 36
   NetSuite 34

H
heap size 13
Hosted Agent 6

L
$LastRunDate
   data filter variable 22
$LastRunTime
   data filter variable 22
List/Record
   NetSuite data type 36
lookups
   NetSuite in mappings 31
   NetSuite in synchronization tasks 23

M
mapping tasks
   NetSuite 31
   NetSuite source properties 31
   NetSuite target properties 31
mappings
   NetSuite 27
   NetSuite lookups 31
   NetSuite source properties 27
   NetSuite target properties 29

N
NetSuite
   administration 7
   changed data capture 6
   configuration for custom NetSuite fields 7
   (continued)
   configuration for saved search fields 7
   configuration for search record field names 7
   connection properties 16
   data types 34
   filters 7
   lookups in mappings 31
   lookups in synchronization tasks 23
   mapping tasks 31
   mappings 27
   rules and guidelines for connections 20
   rules and guidelines in mapping tasks 32
   rules and guidelines in synchronization tasks 25
   sandbox WSDL URL 20
   Source transformation 27
   sources in mapping tasks 31
   sources in mappings 27
   sources in synchronization tasks 22
   supported WSDL URL versions 20
   Target transformation 29
   targets in mapping tasks 31
   targets in mappings 29
   targets in synchronization tasks 23
NetSuite connections
   rules and guidelines 20
NetSuite Connector
   changed data capture 6
   overview 5

S
saved search fields 7, 16
search record field names 7, 16
security tokens 15
Source transformation
   NetSuite properties 27
sources
   NetSuite in mapping tasks 31
   NetSuite in mappings 27
   NetSuite in synchronization tasks 22
   synchronization task schedule for NetSuite sources 24
synchronization tasks
   advanced options for NetSuite 24
   concurrency 24
   concurrency performance 13
   NetSuite lookups 23
   NetSuite source properties 22
   NetSuite source properties 22
   NetSuite target properties 23
   schedule properties for NetSuite 24

T
Target transformation
   NetSuite properties 29
targets
- NetSuite in mapping tasks 31
- NetSuite in mappings 29
- NetSuite in synchronization tasks 23
- synchronization task schedule for NetSuite targets 24
- token-based authentication 15

V
variables
- for data filters 22

W
WSDL URL
- NetSuite sandbox 20
- NetSuite, supported versions 20