Informatica® Cloud Data Integration

NetSuite Connector

Informatica Cloud Data Integration NetSuite Connector
October 2023

Table of Contents

Preface

Use *NetSuite Connector* to learn how to read from or write to NetSuite by using Cloud Data Integration. Learn to create a connection, develop mappings, and run synchronization and mapping tasks in Cloud Data Integration.

Informatica Resources

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

Informatica Documentation

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

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

Informatica Intelligent Cloud Services web site

You can access the Informatica Intelligent Cloud Services web site at http://www.informatica.com/cloud. This site contains information about Informatica Cloud integration services.

Informatica Intelligent Cloud Services Communities

Use the Informatica Intelligent Cloud Services Community to discuss and resolve technical issues. You can also find technical tips, documentation updates, and answers to frequently asked questions.

Access the Informatica Intelligent Cloud Services Community at:

https://network.informatica.com/community/informatica-network/products/cloud-integration

Developers can learn more and share tips at the Cloud Developer community:

https://network.informatica.com/community/informatica-network/products/cloud-integration/cloud-developers

Informatica Intelligent Cloud Services Marketplace

Visit the Informatica Marketplace to try and buy Data Integration Connectors, templates, and mapplets:

https://marketplace.informatica.com/

Data Integration connector documentation

You can access documentation for Data Integration Connectors at the Documentation Portal. To explore the Documentation Portal, visit https://docs.informatica.com.

Informatica Knowledge Base

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

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

Informatica Intelligent Cloud Services Trust Center

The Informatica Intelligent Cloud Services Trust Center provides information about Informatica security policies and real-time system availability.

You can access the trust center at https://www.informatica.com/trust-center.html.

Subscribe to the Informatica Intelligent Cloud Services Trust Center to receive upgrade, maintenance, and incident notifications. The Informatica Intelligent Cloud Services Status page displays the production status of all the Informatica cloud products. All maintenance updates are posted to this page, and during an outage, it will have the most current information. To ensure you are notified of updates and outages, you can subscribe to receive updates for a single component or all Informatica Intelligent Cloud Services components. Subscribing to all components is the best way to be certain you never miss an update.

To subscribe, on the Informatica Intelligent Cloud Services Status page, click **SUBSCRIBE TO UPDATES**. You can choose to receive notifications sent as emails, SMS text messages, webhooks, RSS feeds, or any combination of the four.

Informatica Global Customer Support

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

To find online support resources on the Informatica Network, click **Contact Support** in the Informatica Intelligent Cloud Services Help menu to go to the **Cloud Support** page. The **Cloud Support** page includes system status information and community discussions. Log in to Informatica Network and click **Need Help** to find additional resources and to contact Informatica Global Customer Support through email.

The telephone numbers for Informatica Global Customer Support are available from the Informatica web site at https://www.informatica.com/services-and-training/support-services/contact-us.html.

CHAPTER 1

Introduction to NetSuite Connector

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, targets, and lookups in mappings, synchronization tasks, PowerCenter tasks, and mapping tasks.

You can switch mappings to advanced mode to include transformations and functions that enable advanced functionality.

When you use NetSuite objects in mappings, synchronization tasks, 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 2021_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.

NetSuite Connector assets

Create assets in Data Integration to integrate data using NetSuite Connector.

When you use NetSuite Connector, you can include the following Data Integration assets:

- Data transfer task
- Mapping
- Mapping task
- PowerCenter task
- Synchronization task

For more information about configuring assets and transformations, see *Mappings*, *Transformations*, and *Tasks* in the Data Integration documentation.

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 mapping, synchronization task, 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.

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://webservices.netsuite.com/wsdl/v2019_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:

```
- <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="acctName" 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"/>
```

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:

```
- <complexType name="AccountSearchRowBasic">
  - <complexContent>
    - <extension base="platformCore:SearchRowBasic">
      - <sequence>
          <element name="balance" minOccurs="0" type="platformCore:SearchColumnDoubleField" maxOccurs="unbounded"/>
          <element name="cashFlowRateType" minOccurs="0" type="platformCore:SearchColumnEnumSelectField" maxOccurs="unbounded"/>
          <element name="category1099Misc" minOccurs="0" type="platformCore:SearchColumnSelectField" maxOccurs="unbounded"/>
          <element name="description" minOccurs="0" type="platformCore:SearchColumnStringField" maxOccurs="unbounded"/>
          <element name="externalId" minOccurs="0" type="platformCore:SearchColumnSelectField" maxOccurs="unbounded"/>
          <element name="generalRateType" minOccurs="0" type="platformCore:SearchColumnEnumSelectField" maxOccurs="unbounded"/>
          <element name="internalId" minOccurs="0" type="platformCore:SearchColumnSelectField" maxOccurs="unbounded"/>
          <element name="isInactive" minOccurs="0" type="platformCore:SearchColumnBooleanField" maxOccurs="unbounded"/>
          <element name="name" minOccurs="0" type="platformCore:SearchColumnStringField" maxOccurs="unbounded"/>
          <element name="number" minOccurs="0" type="platformCore:SearchColumnStringField" maxOccurs="unbounded"/>
```

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:

    ```
    [<record 1>]
    <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:

     ```
     [<Object Name>]
     scriptIds = <custom field name1>,<custom field name2>,<custom field name3>
     ```

     For example:

     ```
     [Sales]
     scriptIds = discountPrice,salesDescription,salesEvent
     ```

   - 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>]
     scriptIds = <custom field name1>,<custom field name2>,<custom field name3>
     ```

     For example:

     ```
     [EmployeeSearchAdvanced]
     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>]
     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
  t
  ```

- ```
  [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>]
     <savedSearchId1>=<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:
     ```
     [SavedSearchScriptIdToFieldsMap1]
     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:

```
[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:

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

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

```
<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:

1. Go to
   https://system.netsuite.com/help/helpcenter/en_US/srbrowser/Browser2015_1/script/record/transaction.html.

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,

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

   For example,

   ```
   [SavedSearchScriptIdToFieldsMap1]
   290= userjoin|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,

   ```
   [SavedSearchScriptIdToFieldsMap1]
   290= customSearchJoin|custrecord1423,customSearchJoin|custrecord1421
   ```
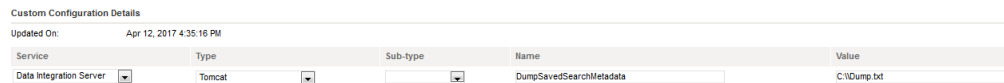
Reading custom record standard fields with custom join

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.

   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:

   ```
   <script Id of custom record>__<standard search column name1>__CSJ,
   <script Id of custom record>__<standard search column name2>__CSJ
   ```

   For example:

   ```
   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:

   ```
   [SavedSearchScriptIdToFieldsMap<version>]
   <savedSearchId1>=CustomSearchJoin|<scriptId of custom record>__<standard field name>
   ```

   For example:

   ```
   [SavedSearchScriptIdToFieldsMap1]
   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:

    ```
    JVMOption6=-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:

6. Restart the Secure Agent.

CHAPTER 2

Connections for NetSuite

Create a NetSuite connection to access NetSuite data. You can use a NetSuite connection to specify sources, targets, and lookups in mappings, synchronization tasks, PowerCenter tasks, and mapping tasks.

Connect to NetSuite

Let's configure the NetSuite connection properties to connect to NetSuite.

Before you begin

Before you configure the connection properties, you'll need to get service URL, account, token ID, and token secret from your NetSuite account.
The following video shows you how to get the information you need:

Connection details

The following table describes the basic connection properties:

Property	Description
Connection Name	Name of the connection. Each connection name must be unique within the organization. Connection names can contain alphanumeric characters, spaces, and the following special characters: _ . + -, Maximum length is 255 characters.
Description	Description of the connection. Maximum length is 4000 characters.
Type	NetSuite

Property	Description
Runtime Environment	The name of the runtime environment where you want to run tasks. Select a Secure Agent, Hosted Agent, or serverless runtime environment. Do not use a Hosted Agent if you use the connection in mappings in advanced mode.

Service URL	NetSuite Web Service Description Language (WSDL) URL to access NetSuite data.
	Consider the following rules for the authentication type that you can use for the WSDL versions:
	- **WSDL version 2016_1 to 2019_2**. You can use username and password or token-based authentication. If you use both, the agent considers the token-based authentication to access Netsuite.
	   Note: Informatica recommends you to use the token-based authentication for a more secure access to NetSuite.
	- **WSDL version 2020_1 and later** - You can use only token-based authentication.
	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 that is specific to your NetSuite account. For more information, see *Configuring NetSuite account-specific service URL*.
	By default, NetSuite connections use version 2021_2_0 of the NetSuite WSDL URL as shown in the following URL:
	`https://webservices.netsuite.com/wsdl/v2021_2_0/netsuite.wsdl`
	Informatica claims support for 2021_2, 2021_1, and 2020_2 WSDLs. You can continue to use WSDL versions older than 2019_2. However, these versions will not receive bug fixes or support.
	You can use NetSuite Connector with the NetSuite 2023_2 sandbox account or release preview account.

Account	NetSuite account ID. To get your account ID, log in to NetSuite, and then click **Setup** > **Integration** > **SOAP Web Services Preferences**.
Token ID	The token ID generated in NetSuite. Applies to token-based authentication.
Token Secret	The token secret generated in NetSuite. Applies to token-based authentication.

Advanced settings

The following table describes the advanced connection properties:

Property	Description
Username	Applicable only when you use the username and password for authentication. User name for a NetSuite account. User name is an email address.
Password	Applicable only when you use the username and password for authentication. Password for the NetSuite account.

Property	Description
Application ID	Optional. NetSuite application ID. 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.

Record Custom Fields	Specify custom NetSuite 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:
	`[<Object Name>] scriptIds = <custom field name1>, <custom field name2>,<custom field name3>`
	For example, `[Sales] 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>] scriptIds = <custom field name1>, <custom field name2>,<custom field name3>`
	For example, `[EmployeeSearchAdvanced]scriptIds = custentity74,custentity66`
	
	- To read 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=custentity_cseg1: select,custentity_csegcs_multsel:multiselect`
	
	- To read data from 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`

Property	Description

Record Filter Fields	Map NetSuite record field names with related NetSuite search record field names so that you can use the fields in filters.
	
	List the record field names and related SearchBasic field names, as follows:
	
	`[<record 1>] <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=nameaddr1=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: [JournalEntry] reversalEntry=memorized`

Saved Search Record 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:
	
	`<savedSearchId1>=<savedSearchDeclaredField1Name>, <savedSearchDeclaredField2Name>,<savedSearchCustomFieldScriptId1>, <savedSearchCustomFieldScriptId2>,<StandardJoin>|<FieldName1>, customSearchJoin|<scriptID1>`
	
	For example, `1000=phone,email,custentity78,custentity65, userJoin| email,customSearchJoin|custrecord1424`
	
	- To read custom segment data, use the following format to add the search custom segment fields:
	
	`[savedSearchId1]=custseg1:select, custseg2:multiselect, custseg3:select...`
	
	For example, `[741]=custseg1:select,custentity_cseg1:select, custentity_csegcs_multsel:multiselect`
	
	- 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:
	
	`<savedSearchId1>=CustomSearchJoin| <scriptId of custom record>__<standard field name>`
	
	For example, `356=CustomSearchJoin|uss_custom_code__internalId`

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 **SUITETALK (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://<Netsuite_account_ID>.suitetalk.api.netsuite.com/wsdl/v2019_2_0/netsuite.wsdl`

Token-based authentication

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.

To use token-based authentication, install an Informatica token-based authentication bundle and generate the 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.

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.
5. Navigate to **Setup** > **Users/Roles** > **Access Tokens** > **New**.
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 need to generate another token in NetSuite. NetSuite does not provide token information for previously generated tokens.

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, target, and lookup 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).

Troubleshoot a NetSuite connection

When you create a NetSuite connection, the following error might 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.

CHAPTER 3

Synchronization tasks with NetSuite

Use a Synchronization task to synchronize data between a source and target.

You can configure a synchronization task using the Synchronization Task wizard.

When you create a task, you can associate it with a schedule to run it at specified times or on regular intervals. Or, you can run it manually. You can monitor tasks that are currently running in the activity monitor and view logs about completed tasks in the activity log.

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:

Property	Description
Connection	Name of the source connection.
Source Type	Select **Single** or **Multiple**.
Source Object	For a single source. Select the source object.
Add Primary	For multiple sources. Select the primary source object.
Add Child	For multiple sources. Displays child objects related to the selected source object. Select a single child object.
Add Sibling	For multiple sources. Displays sibling objects related to the selected source object. Select one or more sibling objects.

Property	Description
Display technical names instead of labels	Displays technical names instead of business names.
Display source fields in alphabetical order	Displays source fields in alphabetical order instead of the order returned by the source system.

NetSuite targets in synchronization tasks

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:

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

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:

Advanced Property	Description
Start Date	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.
End Date	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.
Page Size	Number of rows that the Secure Agent fetches per page. Default is 100 rows. The maximum value is 1000.
Get Deleted	Includes records that were deleted between the start date and end date for changed data capture.
Concurrent Threads	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. **Note:** You cannot use a saved search as a source object in a synchronization task that uses concurrent threads.
	The default value is 1.
Maximum Number of Records to Be Read	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.
Number of Retry	Number of times the agent attempts to execute the request. Default value is 0. **Note:** Retry mechanism works after the transaction starts.
Retry Delay	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.

The following table describes the NetSuite advanced target properties:

Advanced Property	Description
Replace All	Removes existing data from the child object before writing new data to the target.
Batch Size	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.
Success File	Name of the file that contains rows successfully written to the target. The agent writes success log files to the following directory: `<Secure Agent installation directory>\apps\Data_Integration_Server\data\success`
Reject File	Name of the file that contains rows that were not written to the target. The agent writes error log files to the following directory: `<Secure Agent installation directory>\apps\Data_Integration_Server\data\error`
Concurrent Threads	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.

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 14.

CHAPTER 4

Mappings and mapping tasks with NetSuite

Use the Data Integration Mapping Designer to create a mapping. When you create a mapping, you configure a source or target to represent the object.

In advanced mode, the Mapping Designer updates the mapping canvas to include transformations and functions that enable advanced functionality.

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:

Property	Description
Connection	Name of the source connection.
Source type	Select **Single Object** or **Multiple Objects**.
Object	For a single source. Select the source object.
Add Source Object	For multiple sources. Select the primary object.
Add Child Object	For multiple sources. Select the desired child object.

| Property	Description
Add Sibling Objects	For multiple sources. Select the desired sibling objects.
Filter	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.
Sort	Not supported.

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

| Advanced Property	Description
Start Date	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.
End Date	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.
Page Size	Number of rows that the Secure Agent fetches per page. Default is 100 rows. The maximum value is 1000.
Get Deleted	Includes records that were deleted between the start date and end date for changed data capture.
Concurrent Threads	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.
Maximum Number of Records to Be Read	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.
Number of Retry	Number of times the agent attempts to execute the request. Default value is 0. **Note:** Retry mechanism works after the transaction starts.
Retry Delay	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.

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**.
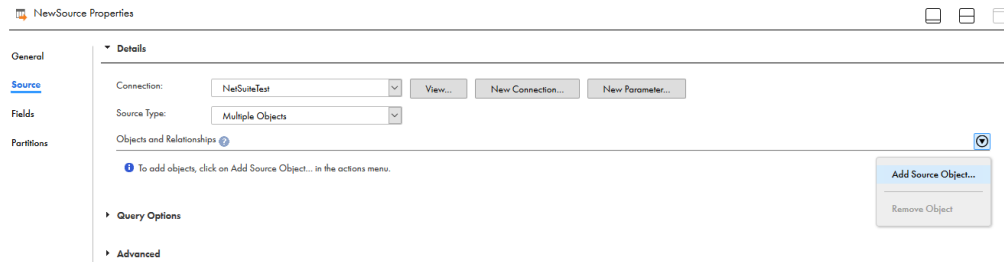
2. Select the primary object.
3. Click the arrow in the primary object's row and from the **Related Object Actions** menu, select either **Add Child Object** or **Add Sibling Objects**.
4. Select the desired child object or one or more sibling objects.

   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:
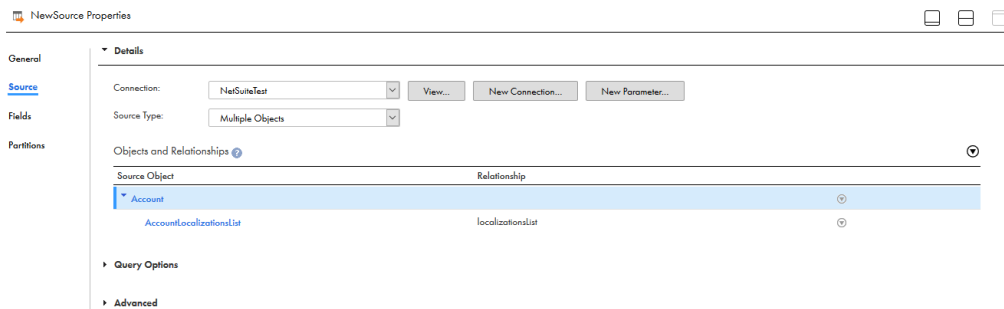
NetSuite targets in mappings

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. You must map at least one source field, in addition to the external ID, from the parent object to the target. Also, you must map at least one source field from the child object to the target. The external ID indicates the parent object that relates to a child object.

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:

Property	Description
Connection	Name of the target connection.
Target Type	Select Single Object or Multiple Objects.
Object	Target object for a single target or primary target object for multiple targets.
Child Object	For multiple targets. Displays child objects related to the selected target object. Select a single child object.
Operation	Target operation.

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

Advanced Property	Description
Replace All	Removes existing data from the child object before writing new data to the target.
Batch Size	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.
Success File	Name of the file that contains rows successfully written to the target.
Error File	Name of the file that contains rows that were not written to the target.
Concurrent Threads	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.
Success File Directory	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: `<Secure Agent installation directory>\apps\Data_Integration_Server\data\success`
Error File Directory	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: `<Secure Agent installation directory>\apps\Data_Integration_Server\data\error`
Forward Rejected Rows	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.

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:

Advanced Property	Description
Start Date	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.
End Date	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.
Page Size	Number of rows that the Secure Agent fetches per page. Default is 100 rows. The maximum value is 1000.
Get Deleted	Includes records that were deleted between the start date and end date for changed data capture.

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:

Advanced Property	Description
Replace All	Removes existing data from the child object before writing new data to the target.
Batch Size	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.
Success File	Name of the file that contains rows successfully written to the target.
Reject File	Name of the file that contains rows that were not written to the target.
Success File Directory	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: `<Secure Agent installation directory>\apps\Data_Integration_Server\data\success`
Error File Directory	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: `<Secure Agent installation directory>\apps\Data_Integration_Server\data\error`

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 read from and write to a NetSuite custom object, ensure that the name of the custom object is not the same as an existing standard object.

- 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 .

NetSuite data type reference

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 transformation and Target transformation 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:

NetSuite Native Data Type	Transformation Data Type	Description
Boolean	Integer	Precision 5, scale 0
Check_box	Integer	Precision 10, scale 0
Currency	String	1 to 104,857,600 characters*
Date	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)
Datetime	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)
Decimal_number	Double	Precision 15

NetSuite Native Data Type	Transformation Data Type	Description
Document	String	1 to 104,857,600 characters*
Double	Double	Precision 15
E-mail_address	String	1 to 104,857,600 characters*
Free-form_text	String	1 to 104,857,600 characters*
Hyperlink	String	1 to 104,857,600 characters*
Inline_HTML	String	1 to 104,857,600 characters*
Int	Integer	Precision 10, scale 0
Integer_number	Integer	Precision 10, scale 0
List/Record	String, Data/Time, Double, Integer	List/Record is treated as a source. Transformations convert List/Record fields to the appropriate transformation data type.
Long	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Multiple_select	String	1 to 104,857,600 characters*
Password	String	1 to 104,857,600 characters*
Percent	String	1 to 104,857,600 characters*
Phone_number	String	1 to 104,857,600 characters*
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*
Time	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)
Time_of_day	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to nanosecond)

* 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