Oracle® Cloud

Using the Oracle NetSuite Adapter with Oracle Integration 3

F45589-06
April 2024

Oracle Cloud Using the Oracle NetSuite Adapter with Oracle Integration 3,

F45589-06

Primary Author: Oracle Corporation

Contents

Preface

1 Understand the Oracle NetSuite Adapter

2 Create an Oracle NetSuite Adapter Connection

Preface

This guide describes how to configure this adapter as a connection in an integration in Oracle Integration.

> **Note:**
>
> The use of this adapter may differ depending on the features you have, or whether your instance was provisioned using Standard or Enterprise edition. These differences are noted throughout this guide.

Topics:

- Audience
- Documentation Accessibility
- Diversity and Inclusion
- Related Resources
- Conventions

Audience

This guide is intended for developers who want to use this adapter in integrations in Oracle Integration.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `https://www.oracle.com/corporate/accessibility/`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `https://support.oracle.com/portal/` or visit `Oracle Accessibility Learning and Support` if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation.

We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

See these Oracle resources:

- Oracle Cloud at `http://cloud.oracle.com`
- *Using Integrations in Oracle Integration 3*
- *Using the Oracle Mapper with Oracle Integration 3*
- Oracle Integration documentation on the Oracle Help Center.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Understand the Oracle NetSuite Adapter

Review the following conceptual topics to learn about the Oracle NetSuite Adapter and how to use it as a connection in integrations in Oracle Integration. A typical workflow of adapter and integration tasks is also provided.

Topics:

Oracle NetSuite Adapter Capabilities

The Oracle NetSuite Adapter enables you to create an integration with a NetSuite application.

NetSuite is a SaaS-based application for business management. The NetSuite platform includes ERP, CRM, PSA, and e-commerce capabilities. To integrate users, NetSuite provides a platform called SuiteCloud that consists of cloud development tools and infrastructure. The SuiteTalk component of the SuiteCloud framework enables integration of NetSuite with other on-premises or cloud solutions.

While SuiteTalk provides the ability to access NetSuite data and business processes through an XML-based API, it requires skills such as Microsoft .NET or Java to build integrations with it. The Oracle NetSuite Adapter addresses these requirements by providing a no-coding approach for building integrations with NetSuite. This enables users who are not professional developers to build integrations with NetSuite.

The Oracle NetSuite Adapter provides the following features:

- Quickly and easily connect on-premises systems and applications with NetSuite.
- Rapidly integrate with both cloud applications and with existing on-premises business systems.
- Automate the process for discovering NetSuite’s web service WSDLs based on the user account.
- Eliminate the need to work with complex polymorphic data objects by elevating NetSuite records.
- Display records based on NetSuite’s categorization.
- Provide available contextual information about business objects and operations to the developer at design time.
- Provide secured invocation to NetSuite’s web services by adhering to the role-based permission structure enforced in NetSuite in a transparent fashion.

- Provide a standard adapter life cycle, controlled runtime environment, and monitoring capabilities.
- Map values of custom fields exposed by any business object while invoking the Oracle NetSuite Adapter. See Map Values for Custom Fields in the Mapper.
- Regenerate integration metadata as you add new custom fields in Oracle NetSuite. This capability eliminates the need to edit an endpoint in an integration. You simply add a new custom field in Oracle NetSuite and regenerate the endpoint for your integration in Oracle Integration by selecting **Refresh endpoints** on the Integrations page. The new custom field is then visible in the mapper for mapping. See Refresh Endpoints for Integrations in *Using Integrations in Oracle Integration 3*.
- Asynchronously invoke of NetSuite APIs. You configure support for asynchronous invocations on the Operations page of the Adapter Endpoint Configuration Wizard. You can also define search criteria on asynchronous invocations. See Invoke Operation Page.
- Expose simple type and complex type custom fields for standard objects in the mapper and during NetSuite endpoint creation for advanced search and saved search operations. Custom field support applies to the following:
  - Basic CRUD operations (except the delete operation). For basic CRUD operations, the custom fields are exposed in the mapper as named fields.
  - Search operations.
  - Synchronous and asynchronous processing modes.

  See Custom Fields Discovery with the Oracle NetSuite Adapter.
- Use the comprehensive search capabilities to search for required records in the NetSuite application. The following types of search are supported.
  - Searching on selected business objects.
  - Searching on selected business objects and the fields of related objects.
  - Searching on selected business objects and the fields of related objects, and defining the fields you require in your response.
  - Invoking a saved search. You can also define the fields you require in your response. You can modify the response columns and add criteria on top of the existing search options.
  - Invoking a saved search without configuring the response fields. This action ensures that the entire response schema is shown in the response mapper for the saved search operation.

  You can also search for custom records.

  You can also define page size and paginate through the search results for any of the search types.

  See Search for Oracle NetSuite Business Objects.
- Perform CRUD operations on custom records in an integration. Custom records are dynamic and customizable building blocks that enable you to create custom fields, lists, and special links with other NetSuite records and transactions. You can use custom records to create a process or functionality that is not available out-of-the-box in NetSuite. The custom records are displayed as elements for mapping in the mapper.

- Provide support for configuring an Oracle NetSuite Adapter connection to use token-based authentication (also known as TBA). Token-based authentication allows you to generate secure, revocable, and nonexpiring tokens for integration clients to use when connecting to Oracle NetSuite. The end user credentials are never exposed and the password does not expire. Because a token is only used by a single application, it provides visibility into which applications are connecting to Oracle NetSuite and control for revoking access. The HMAC_SHA256 algorithm is supported. For existing integrations, you can switch to token-based authentication by updating the connection and re-activating the integration. While updating the connection, you must use the same WSDL as in the existing integration.

Oracle NetSuite Adapter Restrictions

Note the following Oracle NetSuite Adapter restrictions.

- Custom record restrictions:
  - Internal ID detection for standard objects is not supported.
- The Oracle NetSuite Adapter does not discover named custom fields for all standard objects.

> **Note:**
>
> There are overall service limits with Oracle Integration. A service limit is the quota or allowance set on a resource. See Service Limits.

What Application Version Is Supported?

For information about which application version is supported by this adapter, see the Connectivity Certification Matrix.

Oracle NetSuite Adapter Use Cases

Common use cases for the Oracle NetSuite Adapter are as follows:

- Opportunity to order synchronization
- Customer and account data synchronization
- Order management and tracking
- Price and quote configuration in real time

Workflow to Create and Add an Oracle NetSuite Adapter Connection to an Integration

You follow a very simple workflow to create a connection with an adapter and include the connection in an integration in Oracle Integration.

Step	Description	More Information
1	Create the adapter connections for the applications you want to integrate. The connections can be reused in multiple integrations and are typically created by the administrator.	Create an Oracle NetSuite Adapter Connection
2	Create the integration. When you do this, you add trigger and invoke connections to the integration.	Create Integrations and Add the Oracle NetSuite Adapter Connection to an Integration
3	Map data between the trigger connection data structure and the invoke connection data structure.	Map Data in *Using Integrations in Oracle Integration 3*
4	(Optional) Create lookups that map the different values used by those applications to identify the same type of object (such as gender codes or country codes).	Manage Lookups in *Using Integrations in Oracle Integration 3*
5	Activate the integration.	Manage Integrations in *Using Integrations in Oracle Integration 3*
6	Monitor the integration on the dashboard.	Monitor Integrations During Runtime in *Using Integrations in Oracle Integration 3*
7	Track payload fields in messages during runtime.	Assign Business Identifiers for Tracking Fields in Messages and Track Integration Instances in *Using Integrations in Oracle Integration 3*
8	Manage errors at the integration level, connection level, or specific integration instance level.	Manage Errors in *Using Integrations in Oracle Integration 3*

2

Create an Oracle NetSuite Adapter Connection

A connection is based on an adapter. You define connections to the specific cloud applications that you want to integrate.

Topics:

- General Prerequisites for Creating a Connection
- Prerequisites for the Token-Based Authentication Security Policy
- Prerequisites for the User Credentials Security Policy
- Create a Connection
- Upload a Certificate to Connect with External Services
- Refresh Integration Metadata

General Prerequisites for Creating a Connection

To successfully connect to an Oracle NetSuite instance from Oracle Integration, you must fulfill certain prerequisites. This section details the common prerequisites that apply to all types of connections you create with Oracle NetSuite from Oracle Integration.

In addition to these general prerequisites, there are prerequisites specific to each security-policy option available to connect to Oracle NetSuite, which are detailed in the later sections.

Topics:

- Register with Oracle NetSuite and Enable Features
- Assemble the Oracle NetSuite WSDL URL
- Create Custom Fields and Records
- Prerequisites for Using Complex-Type Custom Fields in an Integration

Register with Oracle NetSuite and Enable Features

To connect to Oracle NetSuite, you must have registered with Oracle NetSuite and enabled key features (such as SOAP and REST web services) on your Oracle NetSuite instance.

If you haven't registered with Oracle NetSuite, follow the steps provided here to create an account and enable the required features.

1. Visit http://www.netsuite.com to register with Oracle NetSuite. Ensure that you obtain an account with administrator privileges.
2. Enable connection-related features on your Oracle NetSuite instance.
   a. On your NetSuite home page, select **Setup**, then **Company**, and then **Enable Features**.

b. Click the **SuiteCloud** subtab.

c. In the SuiteScript section, check the following boxes:

   i. **CLIENT SUITESCRIPT**. Click **I Agree** on the SuiteCloud Terms of Service page.

   ii. **SERVER SUITESCRIPT**. Click **I Agree** on the SuiteCloud Terms of Service page.

d. In the SuiteTalk section, check the following boxes:

   i. **SOAP WEB SERVICES**. Click **I Agree** on the SuiteCloud Terms of Service page.

   ii. **REST WEB SERVICES**. Click **I Agree** on the SuiteCloud Terms of Service page.

e. In the Manage Authentication section, check the **TOKEN-BASED AUTHENTICATION** box. Click **I Agree** on the SuiteCloud Terms of Service page.

   You must enable the TBA feature if you want to use the token-based authentication policy to connect to Oracle NetSuite from external applications.

f. Click **Save**.

Assemble the Oracle NetSuite WSDL URL

To create an Oracle NetSuite connection in Oracle Integration, you must have the URL of the Oracle NetSuite's web services description language (WSDL) file.

The general syntax of the Oracle NetSuite WSDL URL is: `https://webservices.netsuite.com/wsdl/<OracleNetSuite_application_version>/netsuite.wsdl`.
Find the version of your Oracle NetSuite instance and assemble the URL corresponding to your instance.

1. On your NetSuite home page, scroll to the bottom of the page to find the version of your Oracle NetSuite instance.

2. Insert the version information into the URL syntax provided earlier.

   If the version of your instance is `2019.1`, your WSDL URL is: `https://webservices.netsuite.com/wsdl/v2019_1_0/netsuite.wsdl`

   For additional information, see NetSuite Versioning and WSDL Versioning Overview.

> **Note:**
>
> To access the Oracle NetSuite Help Center, you must have an Oracle NetSuite account. Once inside the application, you can also access the Oracle NetSuite Help Center by clicking the **Help** link in the upper-right corner of any page.

Create Custom Fields and Records

If you require custom field types and custom record types in Oracle NetSuite for your integration, you must create them in advance in the Oracle NetSuite instance.

For the procedures to create custom field and record types, see Creating a Custom Field and Creating Custom Record Types.

Creation of custom field and record types is a one-time task. After you create them, they are available for selection on the Operations page of the Adapter Endpoint Configuration Wizard.

Prerequisites for Using Complex-Type Custom Fields in an Integration

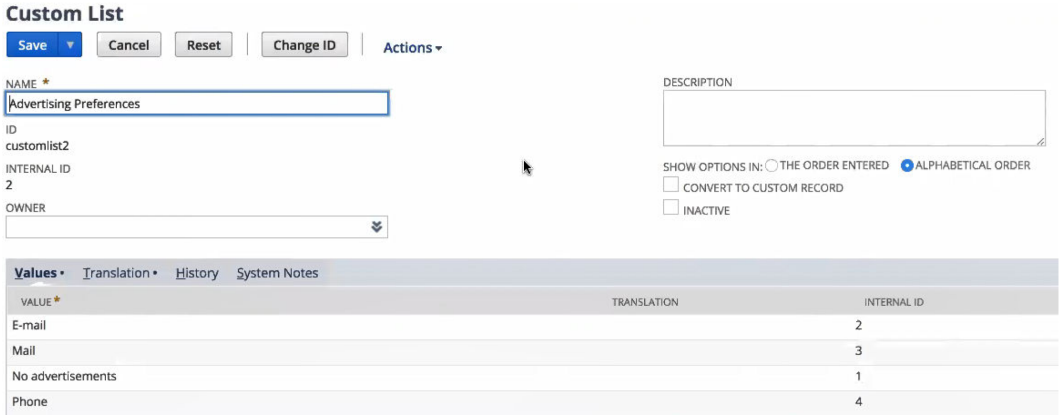
If you want to use a complex-type custom field of Oracle NetSuite in your integration, you must obtain one of the NetSuite Internal IDs associated with the complex field.

You'll require to pass the Internal ID value associated with the field in the data mapper while designing an integration.

> **Note:**
>
> You are not required to pass internal ID values in the data mapper if you're using simple-type custom fields of Oracle NetSuite in your integration flow.

As a reference, here is an example procedure that demonstrates how to find the internal ID values of the constituent list members of an Oracle NetSuite single select or multiselect custom field.

1. On the NetSuite home page, select **Customization**, then **Lists, Records, & Fields**, then **Lists**.
2. On the Custom Lists page, click a link under the **LIST** column to open a custom list. For this example, as shown in the following image, the **Advertising Preferences** custom list is selected. Any of the four values in the **Internal ID** column can be passed to the named custom field's **internalId** element in the mapper.



NAME * Advertising Preferences

ID customlist2

INTERNAL ID 2

OWNER

DESCRIPTION

SHOW OPTIONS IN: ○ THE ORDER ENTERED ● ALPHABETICAL ORDER

□ CONVERT TO CUSTOM RECORD

□ INACTIVE

VALUE *	TRANSLATION	INTERNAL ID
E-mail		2
Mail		3
No advertisements		1
Phone		4

Note:

If you want to use an Oracle NetSuite Adapter connection created prior to the release of support for custom fields, you must first select **Refresh Metadata** from the **Actions** menu ☰ on the Connections page for the connection to use in the integration. See Refresh Integration Metadata. This task is not required for new Oracle NetSuite Adapter connections.

Prerequisites for the Token-Based Authentication Security Policy

If you want to use the token-based authentication (TBA) security policy with the Oracle NetSuite Adapter, you must fulfill the prerequisites specific to this security policy in addition to the general prerequisites.

To create an Oracle NetSuite connection with TBA, you'll require the following details from your Oracle NetSuite instance:

- Consumer Key / Client ID: The key/ID associated with the integration record created for Oracle Integration.
- Consumer Secret / Client Secret: The secret associated with the integration record created for Oracle Integration.
- Token: The token ID associated with the access token created for the Oracle Integration's user account, role, and integration record.
- Token secret: The token secret associated with the access token created for the Oracle Integration's user account, role, and integration record.
- Account ID: Your Oracle NetSuite account ID.

To create and obtain these details, log in to your Oracle NetSuite instance as an **Administrator** and execute the following tasks.

Note:

To perform the TBA-related configuration tasks listed in this section, you must first enable the TBA feature in your Oracle NetSuite account. See Register with Oracle NetSuite and Enable Features.

Topics:

- Create a Role with Token-Based Authentication Permissions
- Create a User Account for Oracle Integration
- Create an Integration Record for Oracle Integration
- Create an Access Token for the User Account
- Make a Note of the NetSuite Account ID

Create a Role with Token-Based Authentication Permissions

Create a new role and assign TBA permissions along with other necessary permissions (specific to your integration) to it. You'll assign the Oracle Integration user account—which you'll subsequently create—to this role.

> **Note:**
>
> As a best practice, avoid using the Administrator and Full Access roles/users in Oracle NetSuite connections that use the TBA security policy.

To create a new role:

1. On the NetSuite home page, select **Setup**, then **User/Roles**, then **Manage Roles**, and then **New**.
2. On the Role page:
   a. Enter a name for the role, for example, `Oracle Integration Role`.
   b. In the **CENTER TYPE** drop-down field, select **System Administrator Center**.
   c. In the Subsidiary Restrictions section, select **All**. For information on subsidiary restrictions, see Restricting Role Access to Subsidiaries.
   d. On the **Permissions** tab, add the required permissions for the role from the four available subtabs: **Transactions**, **Reports**, **Lists**, and **Setup**.

      To add a permission, perform the following actions after selecting any of the subtabs:

      i. Select a permission from the **PERMISSION** drop-down field.
      ii. Select an access level for the permission from the **LEVEL** field.
      iii. Click **Add**.
      To provide TBA permissions to the new role, you must add the **User Access Token** permission to the role with full access. This permission is present on the **Setup** subtab under the **Permissions** tab.

      You can add other permissions to the role depending on the tasks you want to allow the users assigned this role to perform. For any custom role, you must specifically add the SOAP web services permission with the **Full** level. See Assigning the SOAP Web Services Permission to a Role.
   e. After you've added all the necessary permissions, click **Save** to create the new role.

Create a User Account for Oracle Integration

Create a user account for Oracle Integration and assign this account to the TBA role you created previously. You'll use the credentials associated with this user account to connect to NetSuite from Oracle Integration.

If you have already created a user account for Oracle Integration, you can assign the existing account to the new TBA role. See Assign an Existing User Account to a Role. This way, you don't have to consume another open user-account license.

If you require to create a *new* user and assign it to the TBA role, follow the procedure provided here:

1. On the NetSuite home page, select **Lists**, then **Employees**, then **Employees**, and then **New**.
2. On the Employee page:
    a. In the **NAME** fields, enter a first name and last name for the user, for example, `Integration User05`.
    b. In the **EMAIL** field, enter a valid email address.
    c. In the **SUBSIDIARY** drop-down field, select a subsidiary of your choice.
    d. Scroll down and click the **Access** tab to perform additional configurations.
        i. Select the **GIVE ACCESS** and **MANUALLY ASSIGN OR CHANGE PASSWORD** check boxes.
        ii. In the **PASSWORD** field, enter a password for the user account.
        iii. Re-enter the password in the **CONFIRM PASSWORD** field.
        iv. To assign this user to the TBA role created previously:
            - With the **Roles** subtab selected, select the TBA role from the **ROLE** drop-down field; for example, `Oracle Integration Role`.
            - Click **Add**.
    e. Click **Save** to create the new user record.

> **Note:**
>
> To assign an existing Oracle Integration user account to the new role:
>
> 1. From the NetSuite home page, navigate to the Employees page: **Lists**, then **Employees**, and then **Employees**.
> 2. Click **Edit** next to the name of the Oracle Integration user account.
> 3. On the account's page, scroll down and click the **Access** tab.
> 4. With the **Roles** subtab selected, select the role created previously from the **ROLE** drop-down field; for example, `Oracle Integration Role`.
> 5. Click **Add**, and then click **Save**.

Create an Integration Record for Oracle Integration

Before you can create and assign API tokens (for TBA) to a user account, you must create an integration record for the application that will use this user account to access NetSuite.

Create an integration record for the Oracle Integration application.

> **Note:**
>
> If you have already created a TBA-enabled integration record for Oracle Integration, you can skip this section. Reuse the existing record to generate new access tokens. This way, you don't have to maintain multiple integration records and associated consumer keys and secrets for the same application.

1. On the NetSuite home page, select **Setup**, then **Integration**, then **Manage Integrations**, and then **New**.
2. On the Integration page:
   a. Enter a name for the integration record, for example, `Oracle Integration TBA`.
   b. Optionally, enter a description for the record.
   c. Leave the **Enabled** option selected in the **STATE** drop-down field.
   d. On the **Authentication** tab:
      i. Leave the **TOKEN-BASED AUTHENTICATION** check box selected.
      ii. Deselect the **TBA: AUTHORIZATION FLOW** and **AUTHORIZATION CODE GRANT** check boxes.
   e. Click **Save**.

   The confirmation page displays the client credentials for this integration record or application.
3. Note down the **Consumer Key / Client ID** and **Consumer Secret / Client Secret** values. You'll use these credentials to connect to NetSuite from Oracle Integration.

   > **Note:**
   >
   > The system displays the client credentials only the first time you save the integration record. If lose or fail to store these credentials, you'll have to reset the credentials. Edit the integration record and click **Reset Credentials** to generate a new set of client credentials.

Create an Access Token for the User Account

Create and assign an access token to the Oracle Integration user account.

1. On the NetSuite home page, select **Setup**, then **User/Roles**, then **Access Tokens**, and then **New**.
2. On the Access Token page:
   a. In the **APPLICATION NAME** field, select the integration record created previously.
   b. In the **USER** field, select the Oracle Integration's user account.
   c. In the **ROLE** field, select the appropriate TBA role.
   d. Leave the **TOKEN NAME** field unchanged.
   e. Click **Save**.

   The confirmation page displays the token values for the user account.

3. Note down the **Token ID** and **Token Secret** values. You'll use these credentials to connect to NetSuite from Oracle Integration.

> **Note:**
>
> This is the only time the token ID and token secret values are displayed. If lose or fail to store these values, you'll have to create a new token and obtain a new set of values.

Make a Note of the NetSuite Account ID

Along with other credentials, you'll require the NetSuite Account ID to connect to NetSuite from Oracle Integration.

To view your account ID:

1. On the NetSuite home page, select **Setup**, then **Integration**, and then **SOAP Web Services Preferences**.
2. Note down the **Account ID** displayed at the top of the page.
3. Click **Cancel** to exit the page.

Prerequisites for the User Credentials Security Policy

If you want to use the user credentials security policy with the Oracle NetSuite Adapter, you must fulfill the prerequisites specific to this security policy in addition to the general prerequisites.

To create an Oracle NetSuite connection with user credentials, you'll require the following details from your Oracle NetSuite instance:

- Email Address: The email address associated with the user account created for Oracle Integration.
- Password: The password associated with the user account created for Oracle Integration.
- Role: The ID of the role associated with the user account created for Oracle Integration.
- Account ID: Your Oracle NetSuite account ID.
- Application ID: The application ID associated with the integration record created for Oracle Integration.

To create and obtain these details, log in to your Oracle NetSuite instance as an **Administrator** and execute the following tasks.

> **Note:**
>
> To perform the configuration tasks related to the user credentials security policy, you must first enable the SOAP and REST web services in your Oracle NetSuite account. See Register with Oracle NetSuite and Enable Features.

1. Create a new role with web-services permissions. For steps to create a new custom role in Oracle NetSuite, see Create a New Role.
   Along with the necessary permissions specific to your integration, add the following permissions to the new role with full access:
   - REST Web Services
   - SOAP Web Services

   These permissions are present on the **Setup** subtab under the **Permissions** tab.

   > **Note:**
   >
   > As a best practice, avoid using the Administrator and Full Access roles/users in Oracle NetSuite connections that use the user credentials security policy.

2. Create a user account for Oracle Integration and assign this account to the web-services-enabled role you created in the previous step. You'll use the credentials of this user account to connect to NetSuite from Oracle Integration. For steps to create a new user account in Oracle NetSuite, see Create a User Account for Oracle Integration.
   Note down the email address and the password you provide for the user account.

   > **Note:**
   >
   > If you have already created a user account for Oracle Integration, you can assign the existing account to the new web-services-enabled role. See Assign an Existing User Account to a Role. This way, you don't have to consume another open user-account license.

3. Set the default role for the Oracle Integration user account.
   a. On the NetSuite home page, select **Setup**, then **Integration**, and then **SOAP Web Services Preferences**.
   b. On the resulting page:
      i. Select the user account created previously in the **NAME** field.
      ii. In the **WEB SERVICES DEFAULT ROLE** field, select the web-services-enabled role associated with the user account.
      iii. Click **Add**.
      The ID of the role you added is displayed in the row.
      iv. Note down this **Role ID** and also the **Account ID** displayed at the top of the page.
      v. Click **Save**.
4. Create an integration record for the Oracle Integration application. For steps to create a new integration record in Oracle NetSuite, see Create an Integration Record for Oracle Integration.
   a. On the **Authentication** tab in the Integration page, select the **USER CREDENTIALS** check box and deselect the **TOKEN-BASED AUTHENTICATION**, **TBA: AUTHORIZATION FLOW**, and **AUTHORIZATION CODE GRANT** check boxes.
   b. Click **Save**.
   The **APPLICATION ID** is displayed upon saving the record. Note down this ID.

> **Note:**
>
> If you have already created a user-credentials-enabled integration record for Oracle Integration, you can skip this step. Note down the existing record's application ID. This way, you don't have to maintain multiple integration records for the same application.

Create a Connection

Before you can build an integration, you must create the connections to the applications with which you want to share data.

To create a connection in Oracle Integration:

1. In the navigation pane, click **Design**, then **Connections**.
2. Click **Create**.

> **Note:**
>
> You can also create a connection in the integration canvas. See Define Inbound Triggers and Outbound Invokes.

3. In the Create connection panel, select the adapter to use for this connection. To find the adapter, scroll through the list, or enter a partial or full name in the **Search** field.
4. Enter the information that describes this connection.

Element	Description
Name	Enter a meaningful name to help others find your connection when they begin to create their own integrations.
Identifier	Automatically displays the name in capital letters that you entered in the **Name** field. If you modify the identifier name, don't include blank spaces (for example, `SALES OPPORTUNITY`).

Element	Description
Role	Select the role (direction) in which to use this connection (trigger, invoke, or both). Only the roles supported by the adapter are displayed for selection. When you select a role, only the connection properties and security policies appropriate to that role are displayed on the Connections page. If you select an adapter that supports both invoke and trigger, but select only one of those roles, you'll get an error when you try to drag the adapter into the section you didn't select. For example, assume you configure a connection for the Oracle Service Cloud (RightNow) Adapter as only an **invoke**. Dragging the adapter to a **trigger** section in the integration produces an error.
Keywords	Enter optional keywords (tags). You can search on the connection keywords on the Connections page.
Description	Enter an optional description of the connection.
Share with other projects	**Note**: This field only appears if you are creating a connection in a project. Select to make this connection publicly available in other projects. Connection sharing eliminates the need to create and maintain separate connections in different projects. When you configure an adapter connection in a different project, the **Use a shared connection** field is displayed at the top of the Connections page. If the connection you are configuring matches the same type and role as the publicly available connection, you can select that connection to reference (inherit) its resources. See Add and Share a Connection Across a Project.

5. Click **Create**.

   Your connection is created. You're now ready to configure the connection properties, security policies, and (for some connections) access type.

Configure Connection Properties

Enter connection information so your application can process requests.

1. Go to the **Properties** section.
2. In the **WSDL URL** field, specify the URL to use in this integration:

```
https://webservices.netsuite.com/wsdl/NetSuite_application_version/
netsuite.wsdl
```

where *NetSuite_application_version* is the version of the NetSuite application. For example:

```
https://webservices.netsuite.com/wsdl/v2014_2_0/netsuite.wsdl
```

```
https://webservices.netsuite.com/wsdl/v2015_1_0/netsuite.wsdl
```

The web services may or may not be hosted at the above location. The adapter can programmatically determine the correct URL for the web services. NetSuite hosts customer accounts in multiple locations. For example:

- `webservices.netsuite.com`
- `webservices.na1.netsuite.com`

Configure Connection Security

Configure security for your Oracle NetSuite Adapter connection by selecting the security policy and providing the login credentials.

1. Go to the **Security** section.
2. Select the security policy. You must first satisfy several prerequisites. See Create an Oracle NetSuite Adapter Connection.

Security Policy	Fields
User Credentials	• **Email Address** — Enter the email address that serves as the user name. • **Account** — Enter the account. • **Role** — Enter the role ID received from NetSuite, and *not* the role name. Role-based access control ensures that users can only use data and application functionality that is related to their responsibilities. • **Password** — Enter the password. • **Confirm Password** — Reenter the password. • Application Id — Enter the application ID received from NetSuite. This is a mandatory field starting with the 2015_02 version of the NetSuite WSDL.

Security Policy	Fields
Token-Based Authentication	Enter the following values. You must obtain these values before you can complete these fields. See Prerequisites for the Token-Based Authentication Security Policy. The tokens do not expire. If the token is revoked or a new token is generated, you must update the provided values. • **Consumer Key** — Enter the consumer key for the integration record in Oracle NetSuite. • **Consumer Secret** — Enter the consumer secret for the integration record in Oracle NetSuite. • **Confirm Consumer Secret** — Re-enter the consumer secret. • **Token** — Enter the token ID provided by Oracle NetSuite. • **Token Secret** — Enter the token secret provided by Oracle NetSuite. • **Confirm Token Secret** — Re-enter the token secret. • **Account ID** — Enter your Oracle NetSuite account identifier.

Test the Connection

Test your connection to ensure that it's configured successfully.

1. In the page title bar, click **Test**. What happens next depends on whether your adapter connection uses a Web Services Description Language (WSDL) file. Only some adapter connections use WSDLs.

If Your Connection...	Then...
Doesn't use a WSDL	The test starts automatically and validates the inputs you provided for the connection.
Uses a WSDL	A dialog prompts you to select the type of connection testing to perform: • **Validate and Test**: Performs a full validation of the WSDL, including processing of the imported schemas and WSDLs. Complete validation can take several minutes depending on the number of imported schemas and WSDLs. No requests are sent to the operations exposed in the WSDL. • **Test**: Connects to the WSDL URL and performs a syntax check on the WSDL. No requests are sent to the operations exposed in the WSDL.

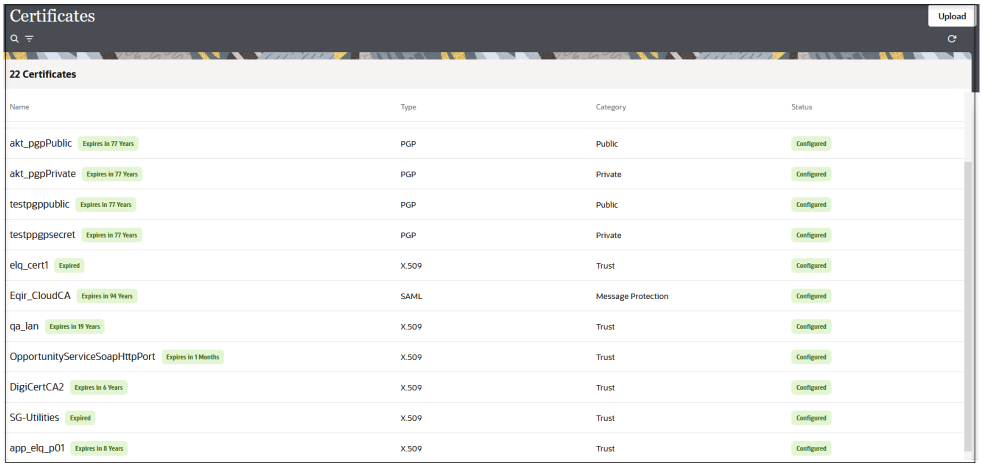
2. Wait for a message about the results of the connection test.
   - If the test was successful, then the connection is configured properly.
   - If the test failed, then edit the configuration details you entered. Check for typos and verify URLs and credentials. Continue to test until the connection is successful.
3. When complete, click **Save**.

Upload a Certificate to Connect with External Services

Certificates allow Oracle Integration to connect with external services. If the external service/endpoint needs a specific certificate, request the certificate and then import it into Oracle Integration.

If you make an SSL connection in which the root certificate does not exist in Oracle Integration, an exception error is thrown. In that case, you must upload the appropriate certificate. A certificate enables Oracle Integration to connect with external services. If the external endpoint requires a specific certificate, request the certificate and then upload it into Oracle Integration.

1. Sign in to Oracle Integration.
2. In the navigation pane, click **Settings**, then **Certificates**.
   All certificates currently uploaded to the trust store are displayed on the Certificates page.
3. Click **Filter** ☰ to filter by name, certificate expiration date, status, type, category, and installation method (user-installed or system-installed). Certificates installed by the system cannot be deleted.



22 Certificates

Name	Type	Category	Status
akt_pgpPublic Expires in 77 Years	PGP	Public	Configured
akt_pgpPrivate Expires in 77 Years	PGP	Private	Configured
testpgppublic Expires in 77 Years	PGP	Public	Configured
testppgpsecret Expires in 77 Years	PGP	Private	Configured
elq_cert1 Expired	X.509	Trust	Configured
Eqir_CloudCA Expires in 94 Years	SAML	Message Protection	Configured
qa_lan Expires in 19 Years	X.509	Trust	Configured
OpportunityServiceSoapHttpPort Expires in 1 Months	X.509	Trust	Configured
DigiCertCA2 Expires in 6 Years	X.509	Trust	Configured
SG-Utilities Expired	X.509	Trust	Configured
app_elq_p01 Expires in 8 Years	X.509	Trust	Configured

4. Click **Upload** at the top of the page.
   The Upload certificate panel is displayed.
5. Enter an alias name and optional description.
6. In the **Type** field, select the certificate type. Each certificate type enables Oracle Integration to connect with external services.
   - Digital Signature
   - X.509 (SSL transport)
   - SAML (Authentication & Authorization)
   - PGP (Encryption & Decryption)
   - Signing key

Digital Signature

The digital signature security type is typically used with adapters created with the Rapid Adapter Builder. See Learn About the Rapid Adapter Builder in Oracle Integration in *Using the Rapid Adapter Builder with Oracle Integration 3*.

1. Click **Browse** to select the digital certificate. The certificate must be an X509Certificate. This certificate provides inbound RSA signature validation. See Implement Digital Signature Validation (RSA) in *Using the Rapid Adapter Builder with Oracle Integration 3*.
2. Click **Upload**.

X.509 (SSL transport)

1. Select a certificate category.
   1. **Trust**: Use this option to upload a trust certificate.
      1. Click **Browse**, then select the trust file (for example, `.cer` or `.crt`) to upload.
   2. **Identity**: Use this option to upload a certificate for two-way SSL communication.
      1. Click **Browse**, then select the keystore file (`.jks`) to upload.
      2. Enter the comma-separated list of passwords corresponding to key aliases.

         > **Note:**
         >
         > When an identity certificate file (`.jks`) contains more than one private key, all the private keys must have the same password. If the private keys are protected with different passwords, the private keys cannot be extracted from the keystore.

      3. Enter the password of the keystore being imported.
   3. Click **Upload**.

SAML (Authentication & Authorization)

1. Note that **Message Protection** is automatically selected as the only available certificate category and cannot be deselected. Use this option to upload a keystore certificate with SAML token support. Create, read, update, and delete (CRUD) operations are supported with this type of certificate.
2. Click **Browse**, then select the certificate file (`.cer` or `.crt`) to upload.
3. Click **Upload**.

PGP (Encryption & Decryption)

1. Select a certificate category. Pretty Good Privacy (PGP) provides cryptographic privacy and authentication for communication. PGP is used for signing, encrypting, and decrypting files. You can select the private key to use for encryption or decryption when configuring the stage file action.
   1. **Private**: Uses a private key of the target location to decrypt the file.
      1. Click **Browse**, then select the PGP file to upload.
      2. Enter the PGP private key password.

b. **Public**: Uses a public key of the target location to encrypt the file.
   i. Click **Browse**, then select the PGP file to upload.
   ii. In the **ASCII-Armor Encryption Format** field, select **Yes** or **No**.
      - **Yes** shows the format of the encrypted message in ASCII armor. ASCII armor is a binary-to-textual encoding converter. ASCII armor formats encrypted messaging in ASCII. This enables messages to be sent in a standard messaging format. This selection impacts the visibility of message content.
      - **No** causes the message to be sent in binary format.
   iii. From the **Cipher Algorithm** list, select the algorithm to use. Symmetric-key algorithms for cryptography use the same cryptographic keys for both encryption of plain text and decryption of cipher text. The following supported cipher algorithms are FIPS-compliant:
      - AES128
      - AES192
      - AES256
      - TDES

c. Click **Upload**.

Signing key

A signing key is a secret key used to establish trust between applications. Signing keys are used to sign ID tokens, access tokens, SAML assertions, and more. Using a private signing key, the token is digitally signed and the server verifies the authenticity of the token by using a public signing key. You must upload a signing key to use the OAuth Client Credentials using JWT Client Assertion and OAuth using JWT User Assertion security policies in REST Adapter invoke connections. Only PKCS1- and PKCS8-formatted files are supported.

1. Select **Public** or **Private**.
2. Click **Browse** to upload a key file.
   If you selected **Private**, and the private key is encrypted, a field for entering the private signing key password is displayed after key upload is complete.
3. Enter the private signing key password. If the private signing key is not encrypted, you are not required to enter a password.
4. Click **Upload**.

Refresh Integration Metadata

You can manually refresh the currently-cached metadata available to adapters that have implemented metadata caching.

Metadata changes typically relate to customizations of integrations, such as adding custom objects and attributes to integrations. There may also be cases in which integrations have been patched, which results in additional custom objects and attributes being added. This option is similar to clearing the cache in your browser. Without a manual refresh, a staleness check is only performed when you drag a connection into an integration. This is typically sufficient, but in some cases you may know that a refresh is required. For these cases, the **Refresh Metadata** menu option is provided.

> **Note:**
>
> The **Refresh Metadata** menu option is only available with adapters that have implemented metadata caching.

1. In the navigation pane, click **Design**, then **Connections**.
2. Hover over the connection to refresh.
3. Click **Actions** •••, then select **Refresh metadata**.

   A message is displayed indicating that the refresh was successful.

3 Add the Oracle NetSuite Adapter Connection to an Integration

When you drag the Oracle NetSuite Adapter into the invoke area of an integration, the Adapter Endpoint Configuration Wizard appears. This wizard guides you through the configuration of the Oracle NetSuite Adapter endpoint properties.

These topics describe the wizard pages that guide you through configuration of the Oracle NetSuite Adapter as an invoke in an integration.

Topics:

- Basic Info Page
- Invoke Operation Page
- Invoke Search Configuration Page
- Summary Page

Basic Info Page

You can enter a name and description on the Basic Info page of each adapter in your integration.

Element	Description
What do you want to call your endpoint?	Provide a meaningful name so that others can understand the responsibilities of this connection. You can include English alphabetic characters, numbers, underscores, and hyphens in the name. You can't include the following characters: • No blank spaces (for example, `My Inbound Connection`) • No special characters (for example, `#;83&` or `righ(t)now4`) except underscores and hyphens • No multibyte characters
What does this endpoint do?	Enter an optional description of the connection's responsibilities. For example: `This connection receives an inbound request to synchronize account information with the cloud application.`

Invoke Operation Page

Enter the Oracle NetSuite Adapter invoke operation values for your integration.

Element	Description
Select a Processing Mode	Select either **Synchronous** or **Asynchronous** as the processing mode. Selecting **Asynchronous** enables you to asynchronously invoke the NetSuite APIs. A help link that provides details about asynchronous processing mode is also displayed.

Element	Description
Select an Operation Type	Select the type of operation to perform on the business object: • **Basic** (for synchronous and asynchronous processing modes) – **Add**: Adds a record into the system. – **Delete**: Deletes a record from the system. – **Get**: Queries the system for a record. – **Update**: Updates an existing record in the system. • **Miscellaneous** Select a specific operation. The **Miscellaneous** option is only available when you select **Synchronous** as the processing mode. – **Attach**: Attaches a file or contact to a business object. * **Select Attach Object Type**: Select **Contact** or **File** to attach an existing contact or file to supported standard records. * **Select Business Object**: Select the object to which to attach the file or contact. – **Detach**: Detaches a file or contact from a business object. * **Select Detach Object Type**: Select **Contact** or **File** to detach it from supported standard records. * **Select Business Object**: Select the object to detach from the file or contact. – **Initialize**: Initializes a record (referred to as a transaction type) by initializing it with values from a related record (referred to as a reference type). * **Select Transaction Type**: Select the transaction type value. * **Select Reference Type**: Select the related reference type value. Only the allowable reference types are displayed. * **Display all Transaction types and Reference types from the schema**: Optionally select to fetch all available transaction and reference types from the schema. Ensure that you select the correct transaction and reference type values from the above two lists. The selected transaction and reference types appear as target elements in the mapper. For example, if your transaction type is **SalesOrder** and reference type is **Opportunity**, you must pass in an **internalid** to the reference type (an existing opportunity for the sales order). You can pass in multiple orders by right-clicking and selecting **Repeat Node** on the **SalesOrder** element. – **GetBudgetExchangeRate**: Gets and filters data related to the budget exchange rate. – **GetConsolidatedExchangeRate**: Gets and filters all data related to the consolidated exchange rate. – **GetCurrencyRate**: Gets and filters the current currency rate. – **GetItemAvailability**: Retrieves the inventory availability for a given list of items.

Element	Description
	– **GetPostingTransactionSummary**: Retrieves a summary of the actual data in an account. • **Search**: Define a search criteria based on the fields of a selected business object. The list of available business objects is refreshed based on your selection. You can define a search criteria on both synchronous and asynchronous invocations. The following types of search criteria are supported: – **Search on selected Business Object.**: Select a business object and define search criteria based on the fields of the object. – **Search on selected Business Object and related objects.**: Search on a business object and the fields of any related business objects. – **Search on selected Business Object and related objects. Also select columns to return.**: Search on a business object and the fields of any related business objects. You can also define the columns (fields) you require in your response. If you select this option and click **Next**, the Search Configuration page is displayed for you to select the response columns you require in your search results. – **Invoke a Saved Search. Also select columns to return.**: Create a saved search that can be invoked. You can modify the response columns and add criteria on top of the existing saved search criteria. If you select this option and click **Next**, the Search Configuration page is displayed for you to select the response columns you require in your search results.
Filter by object name	Type the initial letters to filter the display of business objects.
Select Business Objects	Select the business object to use. Select the business object or custom record type to use. Select **CustomRecord** or **All** to show custom record types. Custom record types are appended with an asterisk.
Your Selected Business Objects	Displays the selected business objects. Displays the selected business objects or custom record types.

Element	Description
Processing Options	Select this link to enable and disable certain aspects of NetSuite cloud application server-side processing when performing an operation. • **Treat Warning As Error**: If selected, the endpoint treats all warning messages that are displayed by the NetSuite cloud application as errors. • **Ignore Read Only Fields**: If selected, the endpoint ignores read-only fields during any requests. • **Return Sublist values in Search Results**: If selected, returns any sublist values contained in the search results. • **Insert record on Update if not Exist**: If selected, a record is inserted if one does not exist. This option is only displayed if the **Update** operation type is selected. • **Search Page Size**: Select the number of search results to return on a single page. The minimum value is **5** and the maximum value is **1000**. For the asynchronous processing mode, the maximum page size value can extend up to **2000**.

Invoke Search Configuration Page

Configure the Oracle NetSuite Adapter search configuration values for your integration.

Topics

- Search Configuration Page
- Search Response Column Selection Page

Search Configuration Page

Select the associated business objects (subobjects) and fields to receive as part of the search operation response.

Element	Description
Primary Business Object	Displays the business object selected on the Operations page.
Select a Saved Search (This element is displayed if you selected **Invoke a Saved Search** on the Operations page.)	Displays the saved searches associated with the selected business object. Select the saved search and click **Test Search** to display the response column results you configured with the business object. Click ⚙ to add more fields to the search. **Note**: Not every response column in the search may be retrieved. If the response does not contain values for a specific response column, that column is not part of the response. Review the results and add any missing response columns as required.
Invoke Saved Search With Response Column As Configured in NetSuite Application	Select to invoke a saved search without configuring the response fields. This action ensures that the entire response schema is shown in the response mapper for the saved search operation. If selected, you cannot add and select response columns (fields).

Element	Description
Click to Add Response Columns	Click to select the response columns for the search to invoke. Custom field columns are also displayed for selection.
Select Response Columns	Displays the response columns (fields) you added by selecting **Click to Add Response Columns**. Click ⚙ to add more fields to the selected response business object.

Search Response Column Selection Page

Select the fields or associated business objects and their fields.

Element	Description
Response Sub Object	Select the basic fields of the business object you selected on the Operations page or select any associated business objects and their fields. Once you select the fields for a business object and click **OK**, that business object is no longer displayed in the **Response Sub Object** list.
Filter by field name	Begin entering letters to filter the display of field names.
Select the Fields	Select the appropriate fields and click **>>**. Custom field are identified by an asterisk.
Your Selected Fields	Displays the selected fields.

Summary Page

You can review the specified adapter configuration values on the Summary page.

Element	Description
Summary	Displays a summary of the configuration values you defined on previous pages of the wizard. The information that is displayed can vary by adapter. For some adapters, the selected business objects and operation name are displayed. For adapters for which a generated XSD file is provided, click the XSD link to view a read-only version of the file. To return to a previous page to update any values, click the appropriate tab in the left panel or click **Go back**. To cancel your configuration details, click **Cancel**.

4

Implement Common Patterns Using the Oracle NetSuite Adapter

You can use the Oracle NetSuite Adapter to implement the following common patterns.

Topics:

- Search for Oracle NetSuite Business Objects
- Map Values for Custom Fields in the Mapper
- Design a Basic Asynchronous CRUD Operation
- Design an Asynchronous Search Operation with Pagination
- Retrieve a Specific Custom Field from a NetSuite Response
- Custom Fields Discovery with the Oracle NetSuite Adapter
- Attach and Detach a Contact with the Oracle NetSuite Adapter

Search for Oracle NetSuite Business Objects

You can configure a NetSuite search operation in the Adapter Endpoint Configuration Wizard.

The following types of search operations are supported:

- Search on a Selected Business Object
- Search on a Selected Business Object and Related Objects
- Search with a Saved Search
- Search on a Selected Business Object and Related Objects and Selecting Columns to Return
- Paginate Through the Search Results

> **Note:**
>
> NetSuite provides documentation about supported mapping values. See SearchDateFieldOperator.

Search on a Selected Business Object

You can perform a search on the fields of a selected business object. The following use case provides an example of how to perform a search operation using the fields of a selected business object in an integration.

- On the Operations page of the Adapter Endpoint Configuration Wizard, **Search on selected Business Object** and the business object (for this example, **Customer** (known as the basic business object)) are selected. This is the simplest search criteria. For this example, no processing options (for example, to return sublist values on search results) and no fields of associated business objects (known as subobjects) of **Customer** are selected, although they can be.

- In the mapper, search criteria can be defined. The fields of the **Customer** business object are displayed in the target tree. There are no associated business objects and fields because none were selected during configuration in the Adapter Endpoint Configuration Wizard.
- You can expand one or more fields and define search criteria by mapping the **searchValue** and operator. Based on the field type, one or more **searchValue** elements and operators can be required. For this example, search criteria are defined using the first name of the **Customer** object and specifying that it must contain **ICSNetSuite**.
- A for-each action can be configured to individually read the search results of customer records (status of search, total records returned, page size, total pages, page index, search ID, and record list).

  The **recordList** is expanded to show that the **Customer** business object is returned. A looping repeating element can be run against **Customer**.

Search on a Selected Business Object and Related Objects

You can search on a selected business object and related objects. This type of search enables you to:

- Search a business object using the associated business object fields as search filters.
- Search a business object using a combination of business object fields and associated business object fields as search filters.

The following use case provides an example of this type of search criteria configuration in an integration.

- On the Operations page of the Adapter Endpoint Configuration Wizard, **Search on selected Business Object and related objects** and the business object (for this example, **Customer**) are selected.
- In the mapper, search criteria can be defined. The fields of the **Customer** (basic) business object and all associated business objects are displayed in the target tree.

  You can define search filters on the basic business object and any associated business objects. For this example, a search is performed on a customer name that includes **ICSNetSuite** with an associated opportunity that has a status of either of two values: **inProgress** or **issuedEstimate**.
- A for-each action can be configured to return the search results of customer records (status of search, total records returned, page size, total pages, page index, search ID, and record list). This is the same response as in the basic search.

  The **recordList** is expanded to show that the **Customer** business object is returned. A looping repeating element is run against **Customer**.

- A log action can be configured to return the first name and last name of the customer and concatenate them. This is the same configuration as in the basic search.

  ```
  concat ( firstName, lastName)
  ```

Search with a Saved Search

You can perform saved searches. This type of search enables you to perform searches that reference:

- An existing saved search.
- An existing saved search in which you override the existing search return columns with new search return columns.
- An existing saved search in which you provide additional search filter criteria on top of the criteria already specified in the saved search.

The following use case provides an example of saved search criteria configuration in an integration.

- On the Operations page of the Adapter Endpoint Configuration Wizard, **Invoke a Saved Search. Also select columns to return** and the business object (for this example, **Customer**) are selected.
- On the Summary page of the Adapter Endpoint Configuration Wizard, the saved searches for the specific business object (**Customer**) are displayed.
- When **Test Search** is clicked, the response column results selected for the business object of this search are verified. This type of configuration enables you to perform a search that references an existing saved search.

  ```
  Success! Note that it is likely that not every Response Column configured
  in the saved search
  would have been retrieved. Hence it is advised to check the below panel
  and manually add any
  missing response columns as required.
  ```

  You can also override the return columns of the existing saved search with new search return columns.

- By selecting **Click to Add Response Columns** on the Search Configuration page, the page is refreshed for selecting additional business objects and their fields. You can also click ⚙ to edit an existing business object and its fields.
- An additional associated business object (**Partner**) and field for that object (**billAddress**) are selected.
- The Search Configuration page shows the new response column that overrides the response columns of the saved search.
  In a saved search, you do not need to define any additional mappings. This means that saved search criteria defined at the application level is used for that particular search.

  In the for-each action, only selected response columns appear in the response. Therefore, the **Partner** business object and **billAddress** field that were added to override the saved search do not appear.

You only define mapping if you want to perform an existing saved search in which you provide additional search filter criteria on top of the criteria already specified in the saved search.

- On the Search Configuration page for the saved search, **Click to Add Response Columns** is selected.
- The **BillingAccount** associated business object and **billingSchedule** and **currency** fields are selected.
- The for-each action shows the newly-added **BillingAccount** associated business object in the list of responses.
- An additional mapper can be added to the integration in which more search criteria can be defined. For example, you can define criteria for the **BillingAccount** associated business object and **billingSchedule** and **currency** fields.

Search on a Selected Business Object and Related Objects and Selecting Columns to Return

You can perform a search on a selected business object and related objects and select the columns to return.

The following use case provides an example of this type of search criteria configuration in an integration.

- On the Operations page, **Search on selected Business Object and related objects. Also select columns to return** and the business object (for this example, **Customer**) are selected.
- On the Search Configurations page, the **Customer** object is displayed as part of the saved search. To specify the response columns to return, **Click to Add Response Columns** is selected.
- The **status** response column is selected for the **Opportunity** associated business object.
- The **Customer** business object and **Opportunity** associated business object are then displayed.
- In the mapper, mappings can be defined on the fields of the **Customer** basic business object and associated subobjects. For this example, fields have been defined on the fields of the **Customer** basic business object.
- A for-each action can be configured to return the response business objects selected on the Search Configuration page. The search results of the **Customer** business object (first name and last name) and **Opportunity** associated business object (status) are displayed.

Paginate Through the Search Results

You can paginate the display of search results for any of the search criteria.

The following use case provides an example of an integration configured to display five records per page of search results.

The **LoopOverPages** while action loops over all the pages. Within the loop, a **LoopOverRecords** for-each action is configured to loop over the records per page of search results.

- On the Operations page of the Adapter Endpoint Configuration Wizard, **Search on selected Business Object** and the business object (for this example, **Customer**) are selected.
- The **Processing Options** link on the Operations page is selected. The **Search Page Size** value is set to **5** to display five records per page of search results.
- In the **InitializeVariables** assign action, three variables must be created and initialized with values (preferably a value of **1**): **PageIndex**, **SearchId**, and **TotalPages**. The **PageIndex** value must be less than or equal to the **TotalPages** value. If not, the Oracle NetSuite Adapter is not invoked and the for-each loop that paginates through the search results does not run.
- A **LoopOverPages** while action is configured to loop over the search results while the **PageIndex** value is less than or equal to the **TotalPages** value.
- In the **LoopOverRecords** for-each action, **totalPages**, **pageIndex**, and **searchId** are used to paginate through the search results.
- In the **AssignVariables** assign action before the loop ends, the **pageIndex** value returned from the search is incremented by one each time. The search ID from the response is mapped to **searchId**. The total number of pages from the response is mapped to **totalPages**.
- In the mapper, the **$pageIndex** and **$searchId** variables must be mapped to the **searchId** and **pageIndex** values that are part of the request.

Map Values for Custom Fields in the Mapper

While invoking the Oracle NetSuite Adapter to create, retrieve, or update any record in the Oracle NetSuite application, you can map values for the custom field types exposed by that particular business object in the mapper.

Overview

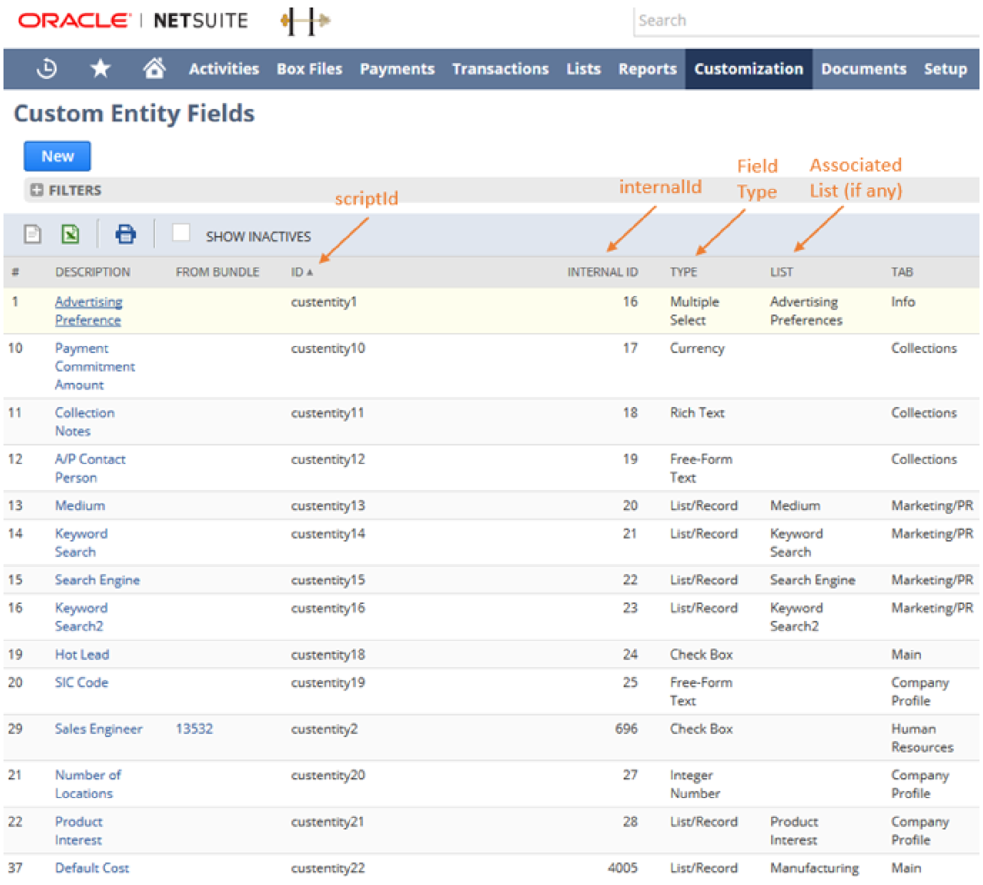
Based on the type of custom field invoked, you can provide the details (**internalId** and **scriptId**) of the custom field being mapped and the value to map to that custom field. For example, **DisplayOrder** from the source schema is being mapped to a custom field defined by the **internalId** of **4567** and **scriptId** of **custentity23**.

Note the following details:

- You can map the value of any custom field you may have added to your business object in Oracle NetSuite.
- Each concrete type extending a **customField** above is a repeating element. Therefore, any number of these types can be mapped (that is, you can repeat **BooleanCustomFieldRef** if you need to map two or more Boolean custom fields).
- The Oracle NetSuite Adapter does not currently discover and show the custom field directly for you to select. Therefore, you must specify **scriptId** and **internalId** for each custom field before mapping its value.
- You can obtain **scriptId** and **internalId** in the Oracle NetSuite application under the **Customization** > **Lists, Records, & Fields** subheading.

Finding internalId and scriptId for a Particular Custom Field

1. Log in to the Oracle NetSuite Application.
2. Look under **Customization** > **Lists, Records, & Fields**.



Custom Entity Fields

#	DESCRIPTION	FROM BUNDLE	ID ▲	INTERNAL ID	TYPE	LIST	TAB
1	Advertising Preference		custentity1	16	Multiple Select	Advertising Preferences	Info
10	Payment Commitment Amount		custentity10	17	Currency		Collections
11	Collection Notes		custentity11	18	Rich Text		Collections
12	A/P Contact Person		custentity12	19	Free-Form Text		Collections
13	Medium		custentity13	20	List/Record	Medium	Marketing/PR
14	Keyword Search		custentity14	21	List/Record	Keyword Search	Marketing/PR
15	Search Engine		custentity15	22	List/Record	Search Engine	Marketing/PR
16	Keyword Search2		custentity16	23	List/Record	Keyword Search2	Marketing/PR
19	Hot Lead		custentity18	24	Check Box		Main
20	SIC Code		custentity19	25	Free-Form Text		Company Profile
29	Sales Engineer	13532	custentity2	696	Check Box		Human Resources
21	Number of Locations		custentity20	27	Integer Number		Company Profile
22	Product Interest		custentity21	28	List/Record	Product Interest	Company Profile
37	Default Cost		custentity22	4005	List/Record	Manufacturing	Main

Finding Which Field Falls Under Which Custom Field Type in the Mapper

The field types you see in the mapper are mapped to the field types you see in the **Type** column of the **Custom Entity Fields** table of the Oracle NetSuite application.

XML Schema Type	Custom Field Type in the Oracle NetSuite Application User Interface
LongCustomFieldRef	Integer
DoubleCustomFieldRef	Decimal Number
BooleanCustomFieldRef	Check Box
StringCustomFieldRef	Free-Form Text Text Area Phone Number E-mail Address Hyperlink Rich Text
DateCustomFieldRef	Date Time of Day or Date/Time (both in one field)
SelectCustomField	List/Record Document

XML Schema Type	Custom Field Type in the Oracle NetSuite Application User Interface
MultiSelectCustomFieldRef	Multiple Select

Mapping Two or More Fields of the Same Type

Use the repeat element functionality in the mapper to map two or more fields of the same type (for example, Boolean).

Handling List Type Fields Such as SelectCustomField and MultiSelectCustomField

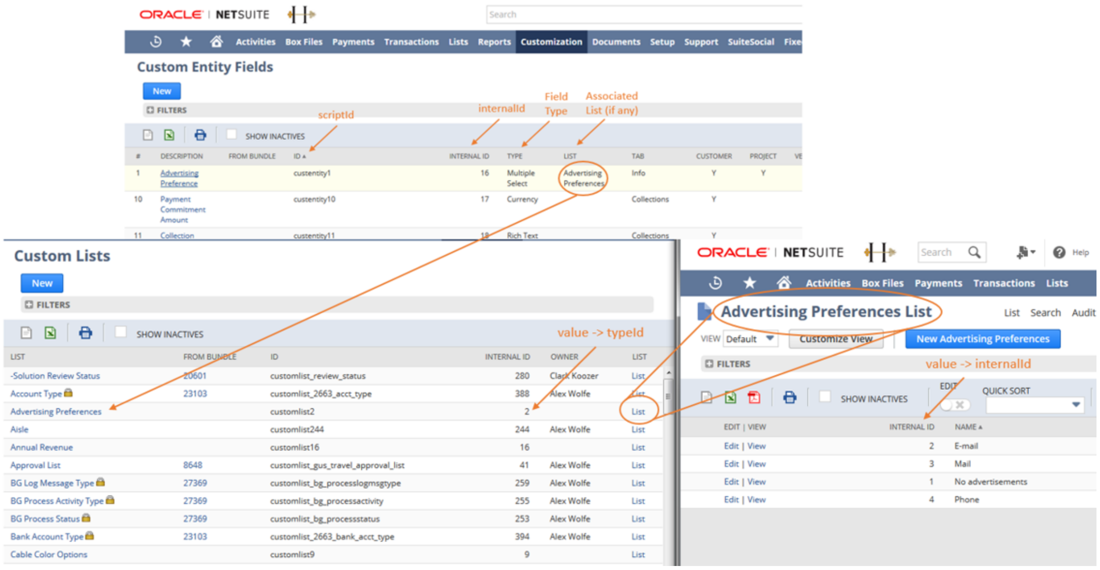
Both **SelectCustomField** and **MultiSelectCustomField** have lists associated with them.

For **SelectCustomField** and **MultiSelectCustomField** in the mapper, the value field is an object type that primarily takes two attributes:

- **typeId**: the **internalId** of the associated list.
- **internalId**: the **internalId** of an item in the associated list.

For **SelectCustomField**, the value element is nonrepeatable because only one item can be selected. For **MultiSelectCustomField**, the value is a repeatable element enabling you to select multiple items.

In the previous example, the **Advertising Preferences** (**internalId** : **16** / **scriptId** : **custentity1**) custom field is mapped and provided with two selections: **Mail** (**internalId** : **3**) and **Phone** (**internalId** : **4**) from the associated **Advertising Preferences** list (**typeId** : **2**). To get these IDs, you open the associated list in the Oracle NetSuite application by going to **Customization** > **Lists, Records, & Fields** > **Lists**, drilling down, and grabbing the **internalId** of each entry in the associated list.



Custom Entity Fields

#	DESCRIPTION	FROM BUNDLE	ID ▲	INTERNAL ID	TYPE	LIST	TAB	CUSTOMER	PROJECT	VE
1	Advertising Preference		custentity1	16	Multiple Select	Advertising Preferences	Info	Y	Y	
10	Payment Commitment Amount		custentity10	17	Currency		Collections	Y		
11	Collection		custentity11	18	Rich Text		Collections	Y		

Custom Lists

LIST	FROM BUNDLE	ID	INTERNAL ID	OWNER	LIST
-Solution Review Status	20601	customlist_review_status	280	Clark Koozer	List
Account Type	23103	customlist_2663_acct_type	388	Alex Wolfe	List
Advertising Preferences		customlist2	2		List
Aisle		customlist244	244	Alex Wolfe	List
Annual Revenue		customlist16	16		List
Approval List	8648	customlist_gus_travel_approval_list	41	Alex Wolfe	List
BG Log Message Type	27369	customlist_bg_processlogmsgtype	259	Alex Wolfe	List
BG Process Activity Type	27369	customlist_bg_processactivity	255	Alex Wolfe	List
BG Process Status	27369	customlist_bg_processstatus	253	Alex Wolfe	List
Bank Account Type	23103	customlist_2663_bank_acct_type	394	Alex Wolfe	List
Cable Color Options		customlist9	9		List

Advertising Preferences List

EDIT | VIEW	INTERNAL ID	NAME ▲
Edit | View	2	E-mail
Edit | View	3	Mail
Edit | View	1	No advertisements
Edit | View	4	Phone

Design a Basic Asynchronous CRUD Operation

This use case provides an overview of how to design an integration in which an Oracle NetSuite Adapter invoke connection is configured with an asynchronous Get operation. The design logic is similar for the other CRUD operations (Add, Delete, Update). Once configured, this integration automatically either submits a new asynchronous job or checks the job status and gets the results based on certain variables being mapped properly.

A high-level overview of the configuration steps is provided below.

1. Create an application or schedule integration. For this example, an application integration is created.
2. Add an adapter as a trigger connection. For this example, the REST Adapter is added and configured.
3. Add an assign action (for this example, named **InitializeVariables**) and initialize the following variables.
   - `jobId` - string data type - value of -`1`
   - `jobStatus` - string data type - value of -`1`
4. Add a while action and specify the following condition.

   ```
   $jobStatus != "finished" and $jobStatus != "finishedWithErrors"
   and $jobStatus != "failed"
   ```

5. Inside the while action, add an Oracle NetSuite Adapter as an invoke connection to make asynchronous calls. This invokes the Adapter Endpoint Configuration Wizard.
   a. On the Operations page, select **Asynchronous** as the processing mode, **Basic** as the operation type, and **GET** as the CRUD value.
   b. Complete the Adapter Endpoint Configuration Wizard.
6. In the mapper positioned in front of the Oracle NetSuite Adapter invoke connection, map the **jobId** variable created in step 3 to the **jobId** variable defined in the Oracle NetSuite Adapter request schema under the **AsyncJobParameters** element.
7. Map additional elements as required for your business use case.
8. Add a second assign action after the Oracle NetSuite Adapter (for this example, named **ReAssignVariables**) and assign the **jobId** and **jobStatus** variables created in step 3 with values from the response of the Oracle NetSuite Adapter invoke connection.
9. Add a switch action and edit the first branch. The loop ends once the following condition is satisfied.

   For the above condition, this results in two routes being created:

   a. **jobStatus** is either **finished**, **finishedWithErrors**, or **failed**. You can now get the results from the Oracle NetSuite Adapter invoke connection response and, based on your business needs, can process the results. For example, for an Add Customer asynchronous job, if the job finished successfully without errors, you can get the **internalId**s of the created Customer records.

b. **jobStatus** is neither of the above values. This means that the asynchronous job is still running. Before you can get the job results, either perform other operations or wait and loop back to the while loop created in step 5.

   The completed integration looks as follows. The Oracle NetSuite Adapter automatically either submits a new asynchronous job or checks the job status and gets the results based on the **jobId** passed in the request.

Design an Asynchronous Search Operation with Pagination

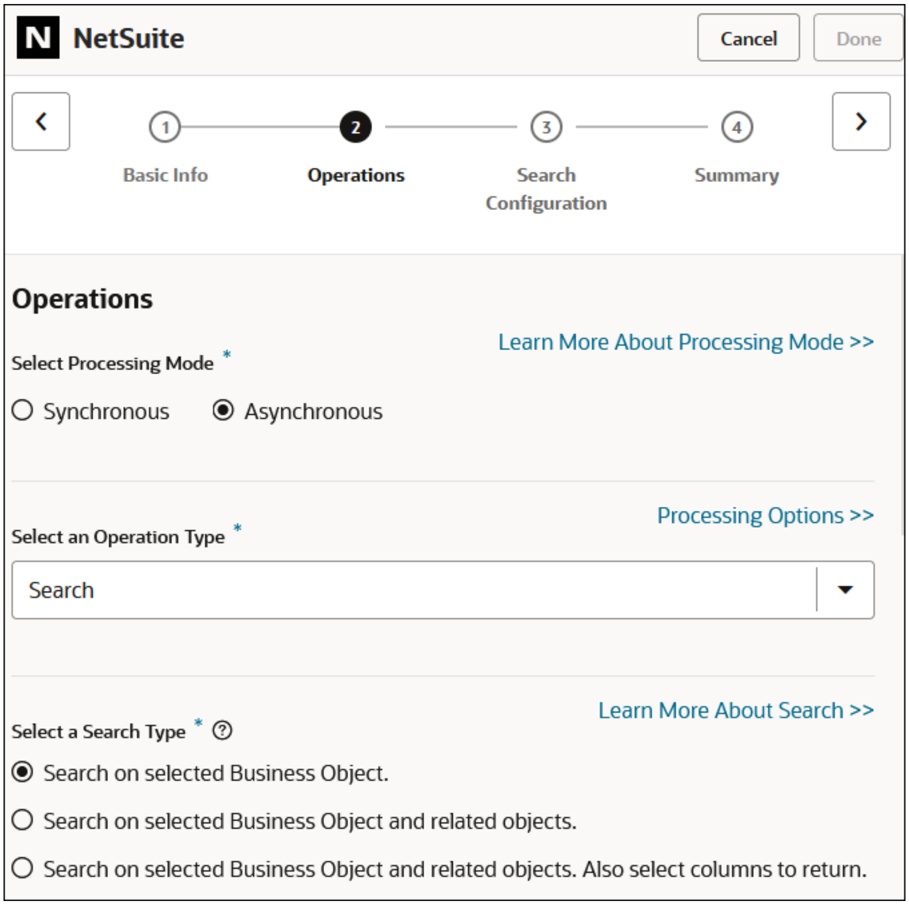
This use case is similar to designing an integration with asynchronous Basic (CRUD) operations. The only difference is that the results returned are paginated. This use case uses an Oracle NetSuite Adapter invoke connection configured to make an asynchronous search operation call.

A high-level overview of the configuration steps is provided below.

1. Create a schedule or application integration. For this example, a schedule integration is created.
2. Add an assign action (for this example, named **InitializeVariables**) and initialize the following variables.
   - `pageIndex` - string data type - value of -`1`
   - `totalPages` - string data type - value of -`1`
   - `jobId` - string data type - value of -`1`
3. Add a while action and specify the following condition.

   ```
   integer($pageIndex) <= integer($totalPages)
   ```

4. Inside the while action, add an Oracle NetSuite Adapter as an invoke connection to make asynchronous calls. This invokes the Adapter Endpoint Configuration Wizard.
5. On the Operations page, select **Asynchronous** as the processing mode, **Search** as the operation type, and **Search on selected Business Object** as the search type.



Operations

Select Processing Mode *

○ Synchronous ● Asynchronous

Select an Operation Type *

Search

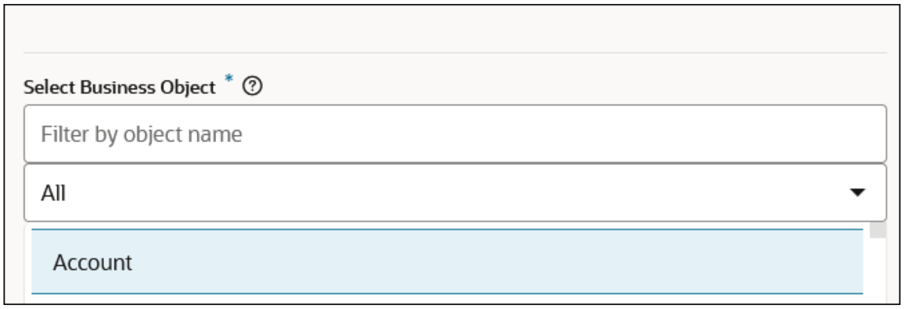
Select a Search Type *

● Search on selected Business Object.

○ Search on selected Business Object and related objects.

○ Search on selected Business Object and related objects. Also select columns to return.

6. Select the business object (for this example, **Account** is selected).



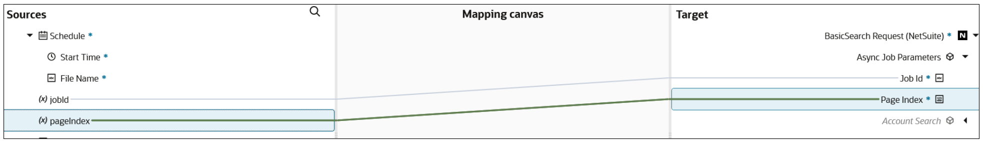
Select Business Object *

Filter by object name

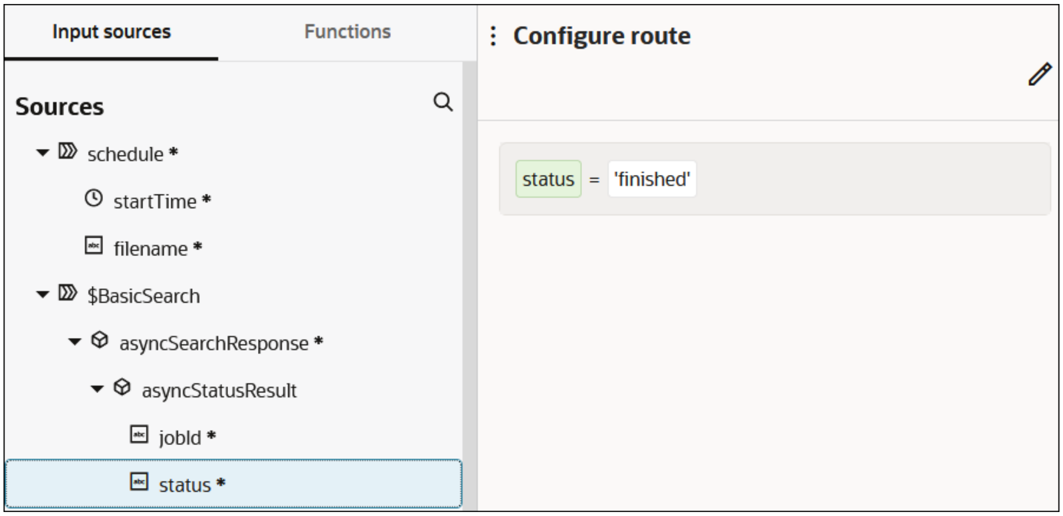
All

Account

7. In the mapper positioned in front of the Oracle NetSuite Adapter invoke connection, map the **jobId** and **pageIndex** variables created in step 2 to the **jobId** and **pageIndex** variables defined in the Oracle NetSuite Adapter request schema under the **Async Job Parameters** element.



8. Map additional elements as required for your business use case.
9. Add a second assign action after the Oracle NetSuite Adapter (for this example, named **ReAssignVariables**) and assign the **jobId** created in step 2 with a value from the response of the Oracle NetSuite Adapter invoke connection.
10. Add a switch action with the condition to check if the **status** of the submitted job is **finished**.



status = 'finished'

For the above condition, this results in two routes being created:

- **status** is **finished**: For this route, create an assign action (for this example, named **IncrementPageIndex**) that increments the **pageIndex** variable and assigns the **totalPages** variable with the actual value from the results of the asynchronous job performed in the Oracle NetSuite Adapter invoke connection. The two assignments needed in this assign action are as follows.
  For the **pageIndex** variable assignment, the Expression Summary is:

  ```
  pageIndex + 1.0
  ```

  For the **totalPages** variable assignment, the Expression Summary is:

  ```
  totalPages
  ```

- **status** is anything other than **finished**: This means that the asynchronous job is either still running, **finishedWithErrors**, or **failed**. Add another switch action in this route to deal with jobs that are **finishedWithErrors** or **failed**. The **otherwise** condition for the new switch action means that the job is still running. In which case, the control loops back to the while loop created in step 3.

When the integration is complete, the Oracle NetSuite Adapter enables you to make use of its extensive search capabilities in asynchronous mode with full support for retrieving the paginated result set.

Retrieve a Specific Custom Field from a NetSuite Response

To retrieve a specific custom field ID from a NetSuite response, you can specify either `internalId` or `scriptId` as an XPath filter along with a value.

This example describes how to map a custom field ID to a target element (for this example, named **ICSEmailId**) with the `scriptId` XPath filter.

1. Open the mapper.
2. Click the target element to access the Build Mappings page to assign a custom field ID (for this example, **ICSEmailId** is selected).
3. Under **customFieldList** in the response schema of the **Source** section, drag the value element of the type of custom field you want to retrieve to the **Statement** section (for this example, the **value** field of **StringCustomFieldRef** is dragged).
4. Click the **Edit** icon to show the **select** section.
5. Copy and paste the entire statement of the **select** section into a text editor such as Notepad.

   ```
   $Netsuite/nsmpr0:getResponse/nsmpr0:CustomerResponse/nsmpr5:
   Customer/nsmpr6:customFieldList/nsmpr5:customField[(fn:resolve-
   QName(@xsi:type, .) =
   fn:QName('urn:core_2018_1.platform.webservices.netsuite.com',
   'StringCustomFieldRef'))]/nsmpr5:value" xml:id="id_58"/>
   ```

6. Add the `{@scriptId]` filter with `[ ]` brackets and assign a custom field ID (for this example, `[@scriptId="custentity23"]` is added).

   ```
   $Netsuite/nsmpr0:getResponse/nsmpr0:CustomerResponse/nsmpr5:
   Customer/nsmpr6:customFieldList/nsmpr5:customField[(fn:resolve-
   QName(@xsi:type, .) =
   fn:QName('urn:core_2018_1.platform.webservices.netsuite.com',
   'StringCustomFieldRef'))]
   [@scriptId="custentity23"]/nsmpr5:value" xml:id="id_58"/>
   ```

7. Copy the updated statement and return to the mapper.
8. Right-click the current statement and select **Input Literal**.
9. Paste the updated statement into the field.
10. Click **Save**.
    The value for the custom field now appears in your response.

Custom Fields Discovery with the Oracle NetSuite Adapter

You can expose simple type and complex type custom fields for standard objects when designing an integration in the mapper and during NetSuite endpoint creation for advanced search and saved search operations with the Oracle NetSuite Adapter. In the mapper, discovered custom fields are available for all search and CRUD operations. During endpoint design time for advanced and saved searches, discovered custom fields are available during the response column selection.

> **Note:**
>
> To ensure that custom field discovery works across different Oracle NetSuite accounts without re-editing the integrations, keep the scriptId consistent across these accounts. This can be ensured by creating the required custom field(s) with the same scriptId across Oracle NetSuite accounts.

- Expose Custom Fields for Standard Objects in the Mapper
- Expose Custom Fields in Advanced Search And Saved Search Operations

Expose Custom Fields for Standard Objects in the Mapper

You can expose simple type and complex type custom fields for standard objects when designing an integration in the mapper with the Oracle NetSuite Adapter.

This use case provides a high-level overview of the design process for the following integration.

> **Note:**
>
> The Oracle NetSuite Adapter does not discover named custom fields for all standard objects.

1. Create an application integration.
2. Add a trigger connection to the integration. For this example, a REST Adapter is added and configured as follows:
   - Perform a POST action on an endpoint.
   - Send a JSON request payload to the endpoint.
   - Configure this endpoint to receive a JSON response payload in return.
3. Add an Oracle NetSuite Adapter connection (for this example, named **NetSuiteUpsertCustomField**) to update (upsert) a custom field.

   > **Note:**
   >
   > To use a connection created prior to the release of support for discovering custom fields, you must first select **Refresh Metadata** from the ☰ menu on the Connections page for this Oracle NetSuite Adapter connection. This task is not required for a new Oracle NetSuite Adapter connection. See Refresh Integration Metadata.

4. Configure the customer request in the mapper between the REST Adapter and Oracle NetSuite Adapter. For this example, a **Customer Update** operation is configured.
   a. Expand the target **Customer** > **customFieldList** > **customField** and note the source **internalId** and **scriptId** elements. To map to the correct custom fields in the mapper,

you must pass in the internal ID value to the **internalId** and **scriptId** elements that you obtained from the NetSuite application described in Prerequisites for Using Complex-Type Custom Fields in an Integration. This is the method for mapping the **customField**. You must map by finding out the **internalId** and **scriptId** values from NetSuite.

b. Click the target elements to add the values to pass. When complete, an **A** is displayed in the **Mapping Canvas**. For this example:

   - The **internalId** value is configured with **4565**.
   - The **scriptId** value is configured with **custentity_cust_priority**. The **scriptId** comes from the NetSuite application.

c. Map to the custom fields for the standard objects, which are displayed one level up under **Customer** > **customFieldList**.
For basic CRUD operations, the custom fields are exposed in the mapper as named fields. Custom field names are derived from the name given to the custom field in NetSuite. This makes mapping easier without needing to know the **internalId** and **scriptId** of a particular custom field for the standard object.
For this example, the mapping is done for a NetSuite update operation. Request mapping is shown from the REST Adapter trigger connection to the NetSuite update operation on the standard objects.

   There are two types of custom fields (both shown in this example):

   - Complex: Enable you to pass in more than one item. For this example, the target complex field **AdvertisingPreference** is shown expanded in the mapper with **1 of 2 ListItem** and **2 of 2 ListItem** for passing in two values. If you expand **ListItem**, the **listItemId** field is displayed. **listItemId** refers to the **internalId** of one of the list members of the complex type custom field. This is not the **internalId** of the **customField**.
   - Simple: Enable you to pass in one item. For this example, the target simple field **ICSEmailId** is shown expanded in the mapper with **nil** for passing in one value.

5. Add a second Oracle NetSuite Adapter (for this example, named **NetSuiteGetCustomField**) to get the customer account number.
6. Configure the mapper between the two Oracle NetSuite Adapters to pass the source **internalId** to the target **internalId**.
7. Configure the mapper after the second Oracle NetSuite Adapter (**NetSuiteGetCustomField**) under the source **CustomerResponse** > **Customer** > **sequence** > **customFieldList** to get the customer response.
Integration design is now complete.

Expose Custom Fields in Advanced Search And Saved Search Operations

You can expose simple type and complex type custom fields during NetSuite endpoint creation for advanced search and saved search operations with the Oracle NetSuite Adapter.

This use case provides a high-level overview of the design process for the following integration:

1. Create a scheduled integration.
2. Add an Oracle NetSuite Adapter connection (for this example, named **NetSuiteAdvancedSearch**) to the integration.
    1. On the Operations page, select **Search on selected Business Object and related objects. Also select columns to return**.
    2. On the Search Configuration page, click the **Click to Add Response Columns** link.
    3. Scroll through the **Select the Fields** list to select the custom fields on which to search, then click **OK** to return to the Search Configuration page. Custom fields are identified by an asterisk.
3. Note that your custom field selection appears at the bottom of the page (for this example, named **PushToSalesforce**).
4. As with the first use case in this section, pass the internal ID value to the **internalId** and **scriptId** elements. For this example, they are located under **CustomerSearchAdvanced** > **customFieldList** > **customField**.
5. Configure the target custom field for the standard object (the configured field is identified by the **A**). For this example, it is displayed one level up under **CustomerSearchAdvanced** > **customFieldList** > **PushToSalesforce**.
6. Configure a for-each action in which the **PushToSalesforce** custom row is selected as the repeating element over which to iterate.
7. Configure the remaining parts of the integration, as necessary. For this example, a logger action is configured.

Integration design is now complete.

Attach and Detach a Contact with the Oracle NetSuite Adapter

You can attach and detach contacts and files in the Oracle NetSuite Adapter. This section provides a use case in which a contact value is first attached to a customer and then detached from that customer.

1. Create an application integration.
2. Add and configure a REST Adapter as a trigger connection.
    1. On the Basic Info page, enter a name.
    2. On the Resource Configuration page, select the **Get** verb.
    3. Select the **Add and review parameters for this endpoint** and **Configure this endpoint to receive the response** configuration options.
    4. On the Request Parameters page, the customer and contact parameters and data types are set as follows:
    5. On the Response page, select **JSON sample** as the response payload format and enter the JSON sample.
    6. Select **response-mapper** from the **Element** list.
    7. Select **JSON** as the media type.

3. Add and configure an Oracle NetSuite Adapter as an invoke connection to attach a contact value to a customer.
    a. On the Basic Info page, enter a name.
    b. On the Operations page, select the **Miscellaneous** operation type, then select **Attach**.
    c. Select **Contact** as the attach object type and **Customer** as the business object to which to attach **Contact** details.
4. In the mapper between the REST Adapter and Oracle NetSuite Adapter, specify concrete values for the target **Customer** > **internalId** element (for this example, `1298`) and the target **Contact** > **internalId** element (for this example, `1307`).
5. Add and configure a second Oracle NetSuite Adapter to detach the contact value from the customer.
    a. On the Basic Info page, enter a name.
    b. On the Operations page, select the **Miscellaneous** operation type, then select **Detach**.
    c. Select **Contact** as the detach object type and **Customer** as the business object from which to detach **Contact** details.
6. In the request mapper between the two Oracle NetSuite Adapters, map the source **Customer** > **internalId** element to the target **Customer** > **internalId** element and set a concrete value of `1307` for the target **Contact** > **internalId** element in the Expression Builder.
7. In the response mapper after the second Oracle NetSuite Adapter, map the source **Customer** > **internalId** element to the target **Customer**. Ensure that you use **@internalId** and do not set a concrete value in the Expression Builder.
8. Invoke the integration. Because this is a REST Adapter-trigger based integration, you can invoke it from the Test page in Oracle Integration). See Test REST Adapter Trigger Connection-Based Integrations in *Using Integrations in Oracle Integration 3*.

5

Troubleshoot the Oracle NetSuite Adapter

Review the following topics to learn about troubleshooting issues with the Oracle NetSuite Adapter.

Topics:

- Oracle NetSuite Summary Type Column in a Saved Search Is Not Synchronizing
- Connection Testing Error Occurs When the Application ID is Not Specified

Oracle NetSuite Summary Type Column in a Saved Search Is Not Synchronizing

If your saved search for the Oracle NetSuite Adapter in the Adapter Endpoint Configuration Wizard includes any field defined in the **Summary Type** column of the **Results** tab of the Oracle NetSuite application, an error occurs that prevents you from proceeding with design.

This is a restriction of the NetSuite SuiteTalk API.

As a workaround, edit the saved transaction search in the Oracle NetSuite application to ensure that no **Summary Type** column (for example, max, min, max, sum, and others) is defined in the **Results** tab. The **Summary Type** column must be blank for all fields.

Connection Testing Error Occurs When the Application ID is Not Specified

If you do not specify a value in the **Application Id** field when configuring the Oracle NetSuite Adapter on the Connections page, the following error is displayed while testing the connection. The **Application Id** field is mandatory. Ensure that you specify a value.

```
CASDK-0005 : A connector specific exception was raised by the application.
Application Id


must be provided for wsdl endpoint versions 2015_2 onwards.
```